THE UNIVERSITY *of* EDINBURGH

From Surfaces to Objects:

Recognizing Objects Using Surface Information and Object Models

Robert Burns Fisher III

PhD
University of Edinburgh
1986

Abstract

This thesis describes research on recognizing partially obscured objects using surface information like Marr's $2\frac{1}{2}$D sketch ([MAR82]) and surface-based geometrical object models. The goal of the recognition process is to produce a fully instantiated object hypotheses, with either image evidence for each feature or explanations for their absence, in terms of self or external occlusion.

The central point of the thesis is that using surface information should be an important part of the image understanding process. This is because surfaces are the features that directly link perception to the objects perceived (for normal "camera-like" sensing) and because surfaces makes explicit information needed to understand and cope with some visual problems (e.g. obscured features). Further, because surfaces are both the data and model primitive, detailed recognition can be made both simpler and more complete.

Recognition input is a surface image, which represents surface orientation and absolute depth. Segmentation criteria are proposed for forming surface patches with constant curvature character, based on surface shape discontinuities which become labeled segmentation boundaries.

Partially obscured object surfaces are reconstructed using stronger surface-based constraints. Surfaces are grouped to form surface clusters, which are 3D identity-independent solids that often correspond to model primitives. These are used here as a context within which to select models and find all object features.

True three-dimensional properties of image boundaries, surfaces and surface clusters are directly estimated using the surface data.

Models are invoked using a network formulation, where individual nodes represent potential identities for image structures. The links between nodes are defined by generic and structural relationships. They define indirect evidence relationships for an identity. Direct evidence for the identities comes from the data properties. A plausibility computation is defined according to the constraints inherent in the evidence types. When a node acquires sufficient plausibility, the model is invoked for the corresponding image structure.

Objects are primarily represented using a surface-based geometrical model. Assemblies are formed from subassemblies and surface primitives, which are defined using surface shape and boundaries. Variable affixments between assemblies allow flexibly connected objects.

The initial object reference frame is estimated from model-data surface relationships, using correspondences suggested by invocation. With the reference frame, back-facing, tangential, partially self-obscured, totally self-obscured and fully visible image features are deduced. From these, the oriented model is used for finding evidence for missing visible model features. If no evidence is found, the program attempts to find evidence to justify the feature is obscured by an unrelated object. Structured objects are constructed using a hierarchical synthesis process.

Fully completed hypotheses are verified using both existence and identity constraints based on surface evidence.

Each of these processes is defined by its computational constraints and are demonstrated on two test images. These test scenes are interesting because they contain partially and fully obscured object features, a variety of surface and solid types and flexibly connected objects. All modeled objects were fully identified and analyzed to the level represented in their models and were also acceptably spatially located.

Portions of this work have been reported elsewhere ([FIS83], [FIS85a], [FIS85b], [FIS86]) by the author.

Table of Contents

List of Figures

List of Tables

Chapter 1

An Introduction to Recognition Using Surfaces

The surface is the boundary between object and non-object and is the usual source and limit of perception. As such, it is the feature that unifies most significant forms of non-invasive sensing, including the optical, sonar, radar and tactile modalities in both active and passive forms. The presence of the surface (including its location) is the primary fact. Perceived intensity is secondary – it informs on the appearance of the surface as seen by the viewer and is affected by the illumination. Given knowledge of the "visible" surfaces of the scene, the identification and location of many objects can be deduced and verified. The development of methods for doing this is the topic of this thesis. Starting from a full surface representation, key issues in the transformation of the scene representation from surfaces to objects are investigated.

Previous research in object recognition has developed theories for recognizing simple objects completely, or complex objects incompletely. Using surface data, the research presented here partially bridges the gap. The main results are:

- Surface information directly provides three dimensional cues for surface detection and grouping, leading to a volumetric description of the objects in the scene.

- Structural properties can be directly estimated from the data, rather than from 2D projections from 3D scenes.

- These properties plus the generic and structural relationships in the model base can be used to directly invoke models to explain the data. This invocation has a formulation suitable for parallel implementation.

- Using surfaces as both the model and data primitive allows direct prediction of visibility relationships, surface matching and verification of identity.

- Moderately complex flexibly connected structures can be completely recognized, spatially located and verified.

This thesis reports the theory and computational constraints behind these points, as implemented in the IMAGINE program. IMAGINE's performance on two test scenes is also reported to substantiate the theoretical results.

1.1 Object Recognition

The following definition is proposed:

> Three dimensional object recognition is the identification of a model structure with a set of image data, such that model-data correspondences are established and the object's three dimensional scene position is known. All features of the model should be fully accounted for – by having consistent image-based evidence supporting either their presence or their absence. The object hypothesis must also be geometrically consistent.

Hence, recognition produces a symbolic assertion about an object, its location and the use of image features as evidence. The matched features must have the correct types, be in the right places and belong to a single, distinct object. Otherwise, though the data might resemble those from the object, the object is improperly assumed and is not at the given location.

Traditional object recognition programs satisfy weaker versions of the above definition. The most common simplification comes from the assumption of a

small, well-characterized, object domain. There, identification can be achieved via discrimination using simply measured image features, such as object color or two dimensional perimeter or the position of a few linear features. This is identification, but not true recognition (i.e. image understanding).

Recognition based on direct comparison between 2D image and model structures – notably through matching boundary sections – has been successful with both grey scale and binary images of flat, isolated, moderately complicated industrial parts. It is simple, allowing geometrical predictions and derivations of object location and orientation and tolerating a limited amount of noise. This method is a true recognition of the objects – all features of the model are accounted for and the object's spatial location is determined.

Some research has started on recognizing 3D objects, but with less success. Model edges have been matched to image edges (in both 2D and 3D) while simultaneously extracting the position parameters of the modeled objects. In polyhedral scenes, recognition is generally complete, but otherwise only a few features are found. The limits of the edge-based approach are threefold:

1. reliable, repeatable and accurate edge information is hard to get from an intensity image,

2. the amount of edge information present in a realistic intensity image is overwhelming and largely unorganizable for matching given current theories, and

3. the edge-based model is too simple to deal with general scenes.

Because of these deficiencies, model-based vision has entered a new phase, addressing these questions (among others):

- what is a good object representation?

- what is a good input data representation?

- how can the models be invoked?

- how can model-to-image correspondences be established?
- how can partial knowledge and constraining relationships be expressed and utilised?
- how can accurate geometrical information be extracted?

Exploiting Surface Data

Early work in machine vision used intensity images as the primary source of scene information. It is now obvious that this representation is too ambiguous locally for direct use in an effective visual system. Even when using edge representations, besides the difficulties of accurately finding the edges, there is still the major problem of interpreting their scene meaning as shadow, reflectance, orientation, highlight or obscuring. For these reasons, researchers are now investigating surfaces and solids as the fundamental visual data representation.

In response, low-level vision research has been working towards direct deduction and representation of scene properties – notably surface depth and orientation. The sources include stereo, optical flow, laser or sonar range finding, surface shading, surface or image contours and various forms of structured lighting.

The most articulated of the surface representations is the $2\frac{1}{2}$D sketch advocated by Marr ([MAR82]). The sketch represents local depth and orientation for the surfaces, and labels detected surface boundaries as being from shape or depth discontinuities. The exact details of this representation and its acquisition are still being researched, but its advantages seem clear enough. These include precise interpretation of all represented quantities in terms of the scene (relative to the viewer), 3D scene information and an accurate geometrical relationship between the image and the scene.

Results suggest that surface information reduces data complexity and interpretation ambiguity ([MAR82]), while increasing the 3D and structure matching

information (e.g. [FAU83]). Other work suggests that a constraint representation and maintenance system is useful for structuring object information, and organizing partial results ([BRO81]). Unfortunately, these vision systems only weakly recognize, through either highly constrained environments or superficial claims to recognition.

The richness of the data in a surface representation, as well as its imminent availability, offers hope for real advances beyond the state of scene analysis summarized above. Distance, orientation and image geometry enable a reasonable reconstruction of the 3D shape of the object's visible surfaces, and the boundaries lead to a figure/ground separation. Because it is possible to segment and characterize the surfaces, more compact symbolic representations are feasible. These symbolic structures would have the same relation to the surface information as edges currently do to intensity information, except that their scene interpretation is unambiguous. If there were:

- reasonable criteria for segmenting both the image surfaces and models,
- simple processes for selecting the models and relating them to the data, and
- an understanding of how all these must be modified to account for factors in realistic scenes (including occlusion),

then object recognition could make significant advances. This is the goal of the research presented in this thesis.

1.2 The Research Problem

The goal of object recognition, as defined in the previous section, is the complete matching of model to image structures, with the concomitant extraction of position information. Hence, the output of recognition is a set of fully instantiated or explained object hypotheses positioned in three dimensions, which are suitable for reconstructing the object's appearance.

The approach investigated requires an object model, which consists, either directly or through subdefinition, of a set of structures (e.g. surfaces) geometrically related in three dimensions. For each model surface, recognition finds those image surfaces that consistently match it, or evidence for their absence (e.g. obscuring structure). The model and image surfaces must agree in location and orientation, and have about the same shape and size, with slight modifications for matching partially obscured surfaces. When surfaces are completely obscured, evidence for their existence comes either from predicting self-occlusion from the location and orientation of the model, or from finding closer, unrelated obscuring surfaces.

The object representation used in this research requires the complete object surface to be segmentable into what would intuitively be considered distinct surface regions. These are what will now be generally called surfaces (either model or data). When considering a cube, the six faces are logical candidates for the surfaces; unfortunately, most natural structures have no such simplicity. This research assumes that object surfaces can be uniquely segmented into regions of roughly constant character, defined by their two principal curvatures. The segmentation assumption presumes the object can be decomposed into rigid structures (possibly flexibly joined), and that segmentation occurs at the necessary scale over the entire surface. It is also assumed that the image surfaces will segment in correspondence with the model surfaces. (If the segmentation criteria is object-based, then the model and data segmentations should be identical.) These assumptions are, of course, unreasonable for the complete solution

to the recognition problem, because surface flexibility and object variations lead to alternative segmentations; however, a start must be made somewhere.

The three models used in the research are: a trash can, a classroom chair, and portions of a PUMA robot. The major common feature of these objects is the presence of regular distinct surfaces uncluttered by shape texture, when considered at a "human" interpretation scale. The objects were partly chosen for experimental convenience, but also to test most of the theories proposed in the thesis. The models are shown in typical views in appendix B. Some of the distinctive features of each object and their implications on recognition are:

- trash can:
  - laminar surfaces – surface grouping difficulties
  - rotational symmetry – surface segmentation and multiple recognitions
- chair:
  - concave surface region (seat back) – new model requirements; surface grouping difficulties
  - thin cylindrical surfaces (legs) – data scale incompatible with model scale
- robot:
  - surface blending – new segmentation relationships
  - flexibly connected subcomponents – unpredictable reference frame relationships and self-occlusions

These objects were viewed in semi-cluttered laboratory scenes that contained both obscured and unobscured views. The test scenes used for analysis are shown in appendix A. Some arrangement of the objects was done to ensure that enough information was present to allow recognition. Because the research made no attempt at solving scale problems, all objects were presented and segmented at appropriate scales.

The images used in the evaluation of this research necessarily required manual processing. Using an intensity image to register all data, nominal depth and surface orientation values were measured by hand. Values at other nearby points in the images were calculated by interpolation. Obscuring and shape segmentation boundaries were selected by hand, which also solved the practical problem of ensuring that the data and model segmentations corresponded. The use of somewhat artificial data obviously avoids problems of segmentation scale, segmentation uniqueness and data errors, but it partly constrains the problem to focus on the primary issues. Otherwise, the artificial data are presumed to be similar to real data. No fully developed processes exist yet to produce the data, but several processes are likely to produce such data soon (chapter 3).

Research Questions

The above discussion summarized the goals and inputs for what was attempted in the research. This section now concludes with a summary of the key questions that were addressed, as related to recognition starting from surfaces:

1. How can the surface data be organized for recognition?
    - What criteria segment surfaces in a viewpoint independent manner?
    - What characterizes surface segments?
2. What is needed in an object representation?
    - What facilitates efficient invocation of object models?
    - What facilitates correspondence with data?
    - What facilitates verification of existence and identity?
3. How can objects be isolated for identification?
    - How can whole surfaces be extrapolated from the observable portions?

- How can the individual surfaces be collected into whole objects?

4. What new 3D object descriptions can be derived from surface information?

   - What new object-centered descriptions are possible?
   - How can these descriptions be obtained?

5. How can the correct model be invoked to explain image data?

   - For what image structures are models invoked?
   - How does evidence accumulate for the hypotheses, and over what information paths?
   - When is there enough evidence for invocation?
   - How does invocation achieve computational efficiency?

6. How can one adequately explain all features of the model?

   - How can the object's 3D position be determined?
   - What image structures are assigned as evidence for model structure?
   - How can missing structure be accounted for properly (i.e. a full explanation for occlusion)?

7. How can one ensure instantiated hypotheses are valid?

   - When is the hypothesized structure physically realizable?
   - When is the image structure consistent with its hypothesized identity?

Recognition is obviously a large problem, and this thesis has attempted to address the issues listed above. A skeletal exploration is appropriate because the use of surface images for recognition is relatively untried. Thus, it is more useful to examine the whole problem from the perspective of surface-based information, exposing the important issues in this recognition paradigm and determining areas for future research, than to exhaustively explore a subtopic of uncertain relevance. Consequently, the results presented in this thesis should not be seen

Figure 1–1: Test Scene 1

as final solutions to the problems introduced above, but as a step on the way towards competent image understanding.

1.3 A Summary of the Research Results

This section summarises the results of the thesis by presenting an example of IMAGINE's surface-based object recognition. The test image discussed in the following example is shown in figure 1–1. This is test scene 1 from appendix A.

Surface Image Inputs

Recognition starts from surface data, as represented in a structure called a labeled, segmented surface image (LSSI). This structure is like Marr's $2\frac{1}{2}$D sketch and includes a pointillistic representation of absolute depth and local surface orientation. The surfaces are separated into regions by boundary segments labeled as shape or obscuring. Shape segmentation is based on orientation, curvature magnitude and curvature direction discontinuities. Obscuring boundaries are placed at depth discontinuities. These criteria segment the surface image into regions of nearly uniform shape, characterized by the two principal curvatures and the surface boundary. As no fully developed processes produce this data yet, the program input is from computer augmented, hand-segmented test images. (Several laboratory systems produce similar data though.) Below shows the input used for the test scene shown in figure 1–1. Figure 1–2 shows the depth values associated with the scene, where the lighter values mean closer points. Figure 1–3 shows the cosine of the surface slant for each image point. Figure 1–4 shows the obscuring boundaries. Figure 1–5 shows the shape segmentation boundaries.

Complete Surface Hypotheses

The image segmentation directly leads to partial or complete object surface segments. Surface completion processes reconstruct obscured portions of surfaces, when possible, by connecting extrapolated surface boundaries behind obscuring surfaces. The advantage of this is twofold – it provides data surfaces more like the original surface for property extraction and the extended surfaces give better image evidence during hypothesis completion. Two processes are used for completing surface hypotheses. The first bridges over gaps in single surfaces and the second links two separated surface patches. Merged surface segments must have roughly the same depth and surface characterization. Figure 1–6 illustrate both

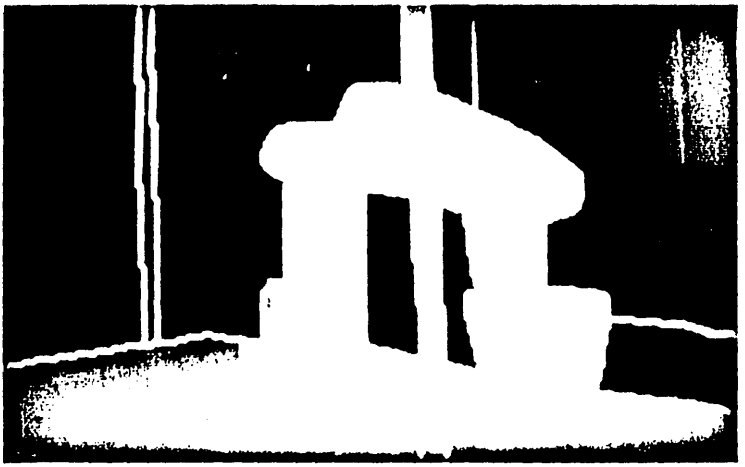


Figure 1–2: Depth Values for Test Scene

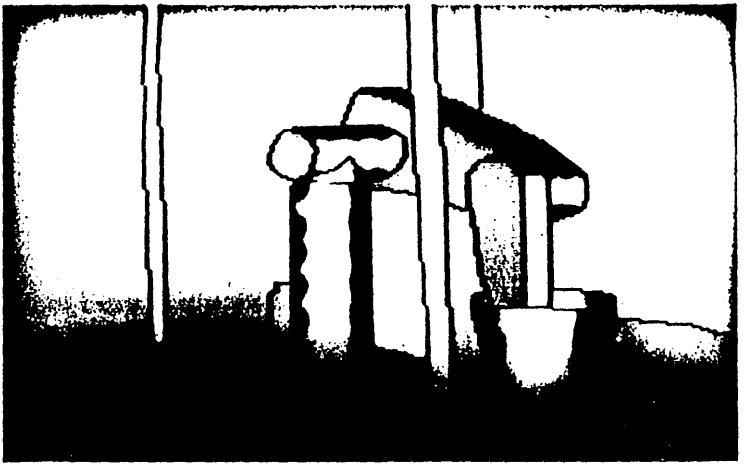


Figure 1–3: Cosine of Surface Slant for Test Scene

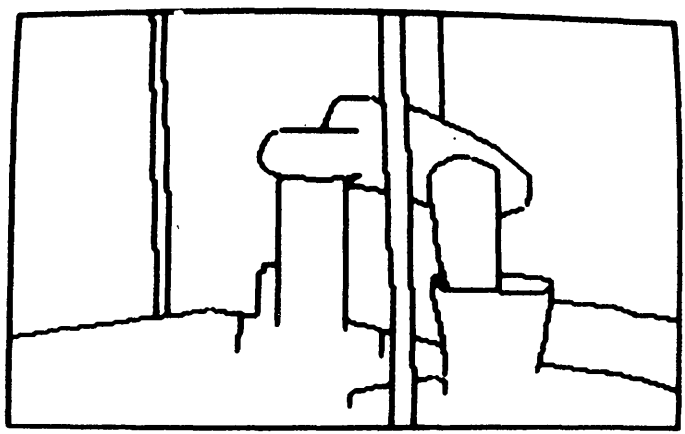


Figure 1–4: Obscuring Boundaries for Test Scene

Figure 1–5: Shape Segmentation Boundaries for Test Scene

rules in showing the original and reconstructed robot upper arm large surface from the test image.

Surface Clusters

Surface hypotheses are joined to form surface clusters, which are blob-like 3D object-centered representations. The goal of this process is to partition the scene into a set of 3D solids, without yet knowing their identities. Surface clusters are useful (here) for aggregating image features into contexts for model invocation and matching. They would also be useful for tasks where identity is not necessary, such as object avoidance.

Forming a surface cluster is based on finding closed loops of isolating boundary segments. Figure 1–7 shows some of the primitive surface clusters for the test scene. The clusters correspond directly with primitive model assemblies. Isolating boundaries are generally obscuring and concave surface orientation discontinuity boundaries. An exception is for laminar objects, where the obscuring boundary across the front lip of the trash can (figure 1–7) does not isolate the surfaces. These criteria determine the primitive surface clusters and larger clusters are formed based on depth ordering relationships.

Surface-Based Object Representation

Objects are compact, connected solids with definable surface boundaries, where the surfaces are rigid and segmentable at some appropriate scale. The objects recognizable by the implemented program may also have rigid subassemblies with possibly flexible interconnections.

Identification requires known object representations with three components: a geometric model, constraints on object properties, and a set of association relationships between objects. Here, the models are designed for object recognition, not image creation, so the represented features are matchable image features.

Figure 1–6: Original and Reconstructed Robot Upper Arm Surface

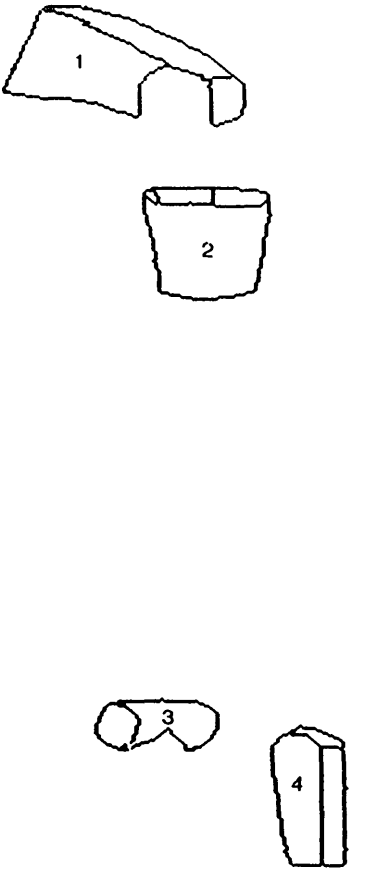


Figure 1–7: Some Surface Clusters for Test Scene 1

The surface patch is the model primitive, because surfaces are the primary data units. This allows direct pairing of data with models, comparison of surface shapes and estimation of model-to-scene transformation parameters. Surfaces are described by their principal curvatures with zero, one or two curvature axes, and by their extent (i.e. boundary). The segmentation ensures that the shape (e.g. principal curvatures) remains relatively constant over the entire surface segment.

Objects are recursively constructed from surfaces or subobjects using coordinate reference frame transformations. Each structure has its own local reference frame transformation and larger structures are constructed by placing the subcomponents in the reference frame of the aggregate. Variable transformations connect subobjects flexibly, by using variables in the attachment relationship. The geometrical relationship between structures is useful for making model to data assignments and for providing the adjacency and relative placement information used by verification.

A portion of the robot model definition is shown below (see chapter 5 for the details).

Illustrated first is the surface definition for the robot upper arm large curved end panel (uendb). The first triple on each line gives the starting endpoint for a boundary segment. The last item describes the segment as a LINE or a CURVE (with its parameters in brackets). PO denotes the segmentation point as a orientation discontinuity point and BO as an orientation discontinuity boundary between surfaces. The next to last line describes the surface type with axis of curvature and radii. The final line gives the surface normal at a nominal point

in the surface's reference frame.

```
SURFACE uendb = PO/(0.0,0.0,0.0) BO/LINE
                PO/(10.0,0.0,0.0) BO/CURVE[0.0,0.0,-22.42]
                PO/(10.0,29.8,0.0) BO/LINE
                PO/(0.0,29.8,0.0) BO/CURVE[0.0,0.0,-22.42]
                CYLINDER [(0.0,14.9,16.75),(10.0,14.9,16.75),22.42,22.42]
                NORMAL AT (5.0,15.0,-5.67) = (0.0,0.0,-1.0);
```

Illustrated next is the rigid upper-arm assembly (upperarm) with its sub-surfaces (e.g. uendb) and the reference frame relationships between them. The first triple in the relationship is the (x, y, z) translation and the second gives the (rotation, slant, tilt) rotation. Translation is applied after rotation.

```
ASSEMBLY upperarm =
                uside AT ((-17.0,-14.9,-10.0),(0.0,0.0,0.0))
                uside AT ((-17.0,14.9,0.0),(0.0,π,π/2))
                uendb AT ((-17.0,-14.9,0.0),(0.0,π/2,π))
                uends AT ((44.8,-7.5,-10.0),(0.0,π/2,0.0))
                uedges AT ((-17.0,-14.9,0.0),(0.0,π/2,3π/2))
                uedges AT ((-17.0,14.9,-10.0),(0.0,π/2,π/2))
                uedgeb AT ((2.6,-14.9,0.0),(0.173,π/2,3π/2))
                uedgeb AT ((2.6,14.9,-10.0),(6.11,π/2,π/2));
```

The assembly that pairs the upper and lower arm rigid structures into a flexibly connected structure is defined now. Here, the lower arm has an affixment

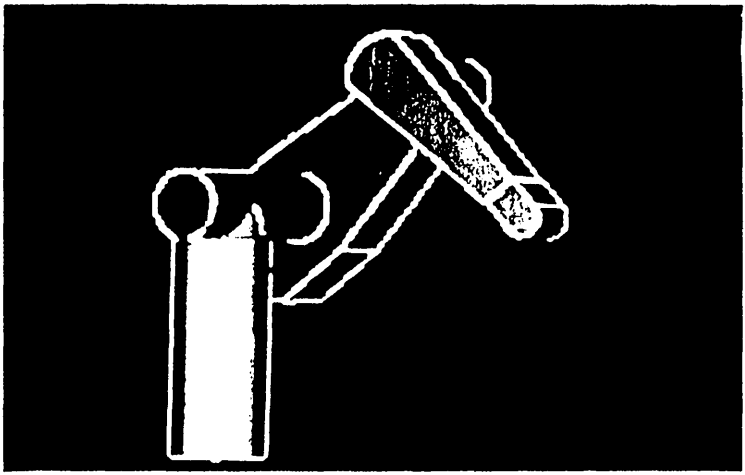


Figure 1–8: Shaded View of Robot Model

parameter that defines the joint angle in the assembly.

```
ASSEMBLY upperasm =
                upperarm AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
                lowerarm AT ((43.5,0.0,0.0),(0.0,0.0,0.0))
                    FLEX ((0.0,0.0,0.0),(jnt3,0.0,0.0));
```

Figure 1–8 shows an image of the whole robot assembly with the surfaces shaded according to surface orientation.

Object property constraints are the basis for direct evidence in the model invocation process and for identity verification. These constraints give the tolerances on attributes associated with the structures, and the importance of the attribute in contributing towards invocation. Some of the constraints associated with the robot base assembly side panel named robbodyside are given below.

The first constraint says that the angle between robbodyside and some adjoining surface should fall in the range 4.5 - 4.9 radians, and the weighting of any evidence meeting this constraint is 0.5. (The weights are ad hoc.)

```
EVIDENCE 4.5 < surface_angle < 4.9 WEIGHT 0.5;
EVIDENCE 0.09 < maximum_surface_curvature < 0.14 WEIGHT 0.5;
EVIDENCE -0.003 < minimum_surface_curvature < 0.01 WEIGHT 0.5;
EVIDENCE 1200.0 < absolute_size < 1600.0 WEIGHT 0.5;
EVIDENCE 1.57 < elongation < 3.5 WEIGHT 0.5;
EVIDENCE 1.17 < boundary_junction_orientation < 1.97 WEIGHT 0.5;
EVIDENCE 45.0 < boundary_length < 55.0 WEIGHT 0.5;
EVIDENCE 0.05 < boundary_curvature < 0.16 WEIGHT 0.5;
```

Association relationships define the network used to accumulate indirect invocation evidence. Between each pair of model structures, several potential relationships exist. The model base defines those that are significant to it by listing the related models, the type of relationship and the strength of association. The definitions related to the robot base assembly side panel robbodyside are given here:

```
SUPERCOMPONENT OF robbodyside IS robbody 0.10;
SUBCOMPONENT OF robbody IS robbodyside 0.90;
```

Evidence for subcomponents comes in visibility groups (i.e. subsets of all object features), because typically only a few of an object's features are visible from any particular viewpoint. While they could be deduced computationally (at great expense), the visibility groups are given explicitly. Those for the upperarm assembly are:

```
SUBCGRP OF upperarm = uside uends uedgeb uedges;
SUBCGRP OF upperarm = uside uendb uedgeb uedges;
```

This says that these are the two significantly different views of the upper arm, and lists the features normally seen in each view. The difference between the two is the visibility of the end panels.

Three Dimensional Feature Description

General identity-independent properties are needed to cue the invocation process; some properties must be extracted before enough evidence exists to suggest the identity of the object, which could then trigger model-directed description processes. Later, these properties are used to ensure that model-to-data surface pairings are correct. The use of 3D information from the surface image makes it possible to compute many object properties directly (as compared to computing them from a 2D projection of 3D data). Most of the properties measured relate to surface patches and include: local curvature, absolute area, elongation and surface intersection angles. Table 1-1 lists the values of these properties for the vertical robot base panel, as estimated from the test image.

Table 1–1: Properties of Robot Base Side Panel From Test Image 1

PROPERTY	ESTIMATED	TRUE
adjacent surface angle	4.8	4.7
adjacent surface angle	2.3	3.1
maximum surface curvature	0.127	0.111
minimum surface curvature	0.0	0.0
absolute area	1238	1413
relative area	1.0	1.0
surface size eccentricity	3.3	2.0
boundary relative orientation	1.78	1.57
boundary relative orientation	1.38	1.57
number of parallel boundaries	2	2
boundary curve length	27.3	28.2
boundary curve length	46.1	50.0
boundary curve length	51.1	50.0
boundary curvature	0.038	0.11
boundary curvature	0.011	0.0
boundary curvature	0.010	0.0

Model Invocation

Model invocation is necessary because of the many potential identities for any image structure, and because generic representation requires suggestive indexing (i.e. there may not be an exact model for the data). Invocation is based on plausibility, rather than certainty, and this notion is expressed through accumulating various types of evidence for objects in an associative network. When the plausibility of a structure having a given identity is high enough, a model is invoked.

Plausibility accumulates from direct and indirect component and generic evidence. Evidence accumulation allows graceful degradation from erroneous data. Direct evidence is obtained when properties of the structure (acquired in the description process) satisfy constraints expressed in the object model. Each relevant description contributes direct evidence in proportion to a weight factor (emphasizing its importance) and the degree that the evidence fits the constraints. When the data values from table 1–1 are associated with the constraints given previously, the resulting direct evidence plausibility for the robot base side panel in the test image is 0.056 in the range [-1.0,1.0]. The low value arises mainly because the boundaries at the upper end of the cylinder were corrupted in the data. Nonetheless, there is positive direct evidence for the identity.

Indirect evidence arises from conceptual associations with other structures and identifications. In the test example, the most important associations are supercomponent and subcomponent, because of the structured nature of the objects. Robot upper arm assemblies are linked to whole robot arm assemblies by these component links. Generic associations are also used (but not in the robot example): a specific type of office chair is linked with a generic office chair. Inhibitory evidence comes from competing identities. The associations related to the robot base side panel were given above. All evidence types are combined to give an integrated evidence value. The evidence for the robbodyside model was:

- direct properties: 0.056

- supercomponent (robbody): 0.254

- inhibition: none

and the accumulated value was 0.081. No inhibition was received because there were no competing identities with sufficiently high plausibility. The supercomponent evidence is scaled (by 0.1) because it is only coincidental. The presence of the supercomponent implies the subcomponent is present somewhere, but not that this particular image structure is that subcomponent.

Plausibility is only associated with the structure being considered and its context; otherwise, models would be invoked for unlikely structures. In other words, invocation must localize its actions to some context inside which all relevant data and structure must be found. Image surface hypotheses are the context for model surfaces and surface clusters are the context for model assemblies. The most plausible context for invoking the upper arm assembly model in the test image is shown as blob 1 in figure 1–7, which is the true upperarm.

The invocation computation is based on accumulating plausibility in a relational network of context × identity nodes linked to each other by the indirect association links and linked to the data by the direct association constraints. The lower level nodes in this network are general object structures, such as corners, planar surfaces, common curves, or right angle surface junctions. From these, higher level, object structures are linked hierarchically. In this way, plausibility accumulates upwardly from simple to more complex structures. This structuring provides both richness in discrimination through added detail, and efficiency of association (i.e. a structure need link only to the most compact levels of subdescription, not to subdescriptions beneath these). Though every model must ultimately be a candidate for every image structure, the network formulation achieves efficiency through judicious selection of appropriate conceptual units and computing plausibility over the entire network in parallel. This network must be restructured for each image, but a proposal for how this can be done dynamically is given in section 9.3. Figure 1–9 shows a portion of the network structure for the trashcan model.

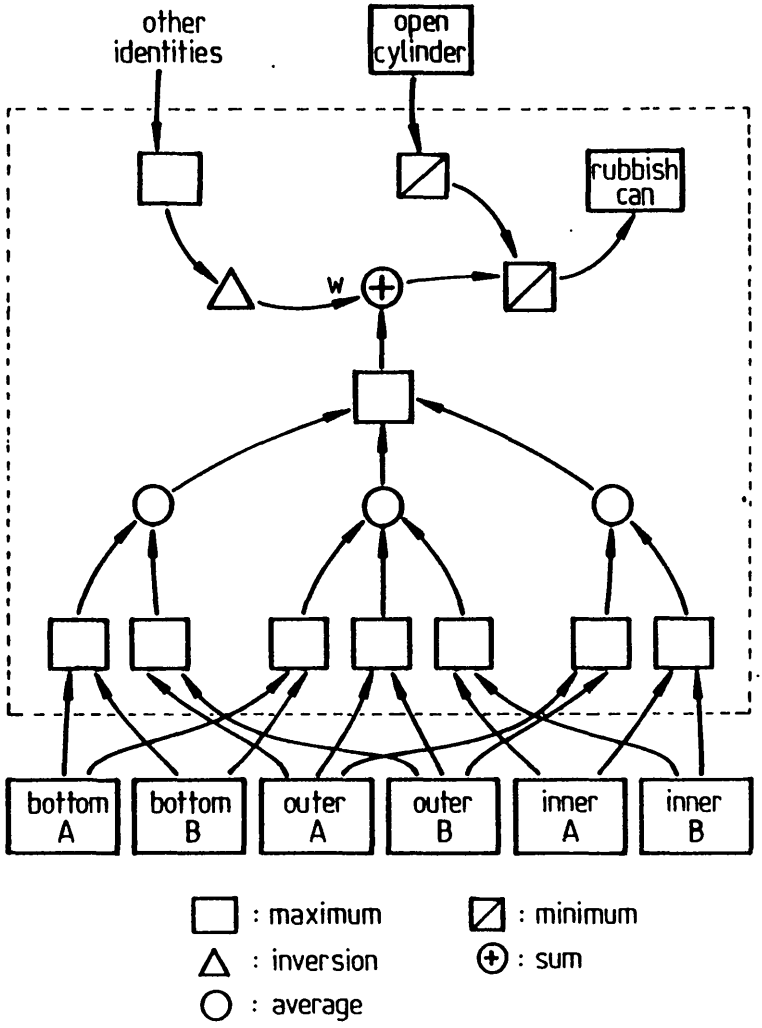


Figure 1-9: Fragment of Invocation Network for Trashcan Assembly

Hypothesis Completion

Full object recognition requires finding image evidence for all model features, which are surfaces and recursively defined subcomponents (by the modeling assumptions). Invocation provides the model-data correspondences for forming the initial hypothesis, which is used for estimating the 3D location and orientation. Invocation thus eliminates most substructure search by directly pairing features. All other data must come from within the local surface cluster context.

Hypothesis completion requires global location and orientation estimates. The spatial relationships between structures are constrained by the geometrical relationships of the model and inconsistent data implies an inappropriate invocation or feature pairing. Object orientation is estimated by mapping the nominal orientations of pairs of model surface vectors to corresponding image surface vectors. Pairs are used because a single vector allows a remaining degree of rotational freedom. Surface normals and curvature axes are the two types of surface vectors used. Translation is estimated from the allowable range of oriented model surfaces consistent with the image data.

Because of data errors, the six degrees of spatial freedom are represented as parameter ranges. Each new model-data feature pairing contributes new spatial information, which helps further constrain the parameter range. Previously recognized substructures also constrain object position.

Table 1–2 lists the measured and estimated location positions, orientation angles and flexible attachment angles for the robot in test image 1. This data was obtained from an image taken at about 500 cm. As can be seen, the translations were estimated well, but the rotations were more inaccurate. This was because of:

- insufficient surface evidence to better constrain the position of individual assemblies, and
- inadequacies in the geometrical reasoning method, when integrating multiple assemblies.

Table 1–2: Measured And Estimated Spatial Parameters

PARAMETER	MEASURED	ESTIMATED
X	488 (cm)	486 (cm)
Y	89 (cm)	85 (cm)
Z	554 (cm)	552 (cm)
Rotation	0.0 (rad)	0.242 (rad)
Slant	0.793 (rad)	0.904 (rad)
Tilt	3.14 (rad)	3.64 (rad)
Joint 1	2.24 (rad)	2.29 (rad)
Joint 2	2.82 (rad)	3.07 (rad)
Joint 3	4.94 (rad)	4.34 (rad)

A variety of model-driven processes contribute to completing an oriented hypothesis once position is estimated. They are, in order:

1. decide back-facing surfaces
2. decide tangential surfaces
3. predict visibility of remaining surfaces
4. search for missing visible surfaces
5. bind rigidly connected subobjects
6. bind flexibly connected subobjects
7. explain some incorrectly segmented surfaces
8. validate externally obscured structure

Hypothesis completion has a "hierarchical synthesis" character, where data surfaces are paired with model surfaces, surface groups are matched to assemblies and assemblies are matched to larger assemblies. The three key constraints

on the matching are: (1) localization in the correct image context (i.e. surface cluster), (2) correct feature identities and (3) consistent reference frame relationships.

Adding a new surface or a rigidly connected subcomponent requires meeting only the above three requirements. Joining together two flexibly connected assemblies also gives the values of the variable attachment parameters by unifying the respective reference frame descriptions. The parameters must also meet any specified constraints, such as on joint angles in the robot model.

The construction process tries to find evidence for every portion of the model. Many features are paired during the invocation process. Others, such as the back of the trash can in the test image, need to be paired by a model-directed process. Given the oriented model, the image positions of unmatched surfaces can be predicted. Then, any surfaces in the general area that:

- have not already been previously used,
- belong to the surface cluster and
- have the correct shape and orientation

can be used as evidence for the unpaired model features. Later verifications ensure that correct pairings were made.

Missing structure requires understanding the three cases of occlusion, predicting or detecting its occurrence and showing that the image data is consistent with the expected visible portion of the model. The easiest case of back-facing and tangent surfaces can be predicted using the orientation estimates with known observer viewpoint and the surface normals deduced from the geometrical model. A raycasting technique (i.e. predicting an image from an oriented model) handles self-obscured front-facing surfaces by predicting the location of obscuring surfaces and hence which portions of more distant surfaces are invisible. The final case occurs when unrelated structure obscures portions of the object. Assuming enough evidence is present to invoke and orient the model, occlusion

Table 1-3: Predicted Trash Can Visibility

SURFACE	VISIBLE PIXELS	OBSC'D PIXELS	TOTAL PIXELS	VISIBILITY
outer front	1479	8	1487	full
outer back	1	1581	1582	back-facing
outer bottom	5	225	230	back-facing
inner front	0	1487	1487	back-facing
inner back	314	1270	1584	partial-obsc
inner bottom	7	223	230	full-obsc

can be confirmed by finding closer unrelated surfaces responsible for the missing image data.

The self-occlusion visibility analysis for the trash can in the scene is given in table 1-3. The results are correct. Minor prediction errors occur at edges where surfaces do not meet perfectly. Figure 1-10 shows the boundaries of the found portions of the robot model as predicted by the orientation parameters and superposed over the original intensity image. No hidden line removal was used. Because of minor cumulative rotation angle errors from the robot's base position, the position of the lower arm is somewhat away from its observed position. However, when it was initially recognized, its position was closer. Further, the picture shows that the global understanding is correct. In analysis, all features were correctly paired, predicted invisible or verified as externally self-obscured. The numerical results in table 1-2 also show good performance.

Identity Verification

The final step in the recognition process is verification. Verification ensures that instantiated hypotheses are valid physical objects and have the correct identity (i.e. have all object properties). This is necessary because model invocation suggests a particular object, which then acquires rough model instantiation and

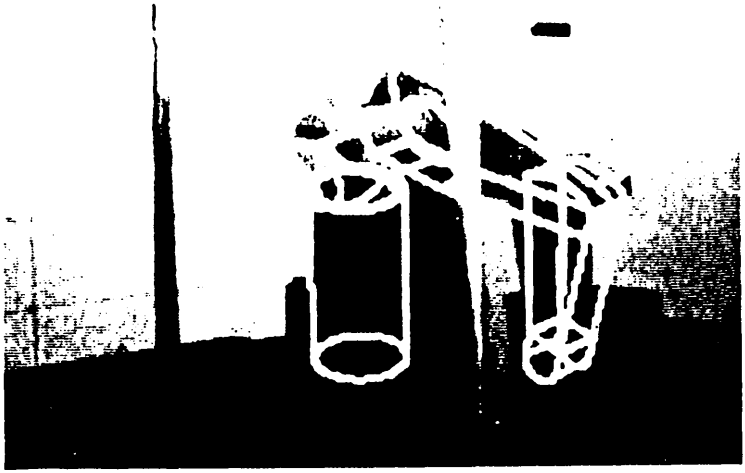


Figure 1–10: Predicted Surface Boundaries for Found Robot Assembly

orientation. It is necessary to verify the details for correctness. A proper, physical, object is more certain if all surfaces are connected and they enclose the object. Correct identification is more likely if all model features are accounted for, the model and corresponding image surface shapes and orientations are the same, and the model and image surfaces are connected similarly. The constraints used to ensure correct surface identities in the test image were:

- has approximately correct size
- has approximately correct surface shape

For solids they were:

- has no duplicated use of image data
- all predicted back-facing surfaces have no data
- all adjacent model surfaces are adjacent in data

- all subfeatures have correct orientation

- all features predicted as partially self-obscured during raycasting are observed as such (i.e. have appropriate occluding boundaries)

In the example given above, all correct object hypotheses passed these constraints. The only spurious structures to pass verification were very similar to the invoked model or symmetric subcomponents.

Discussion

This recognition process was clearly successful on the test image. However, much research is still needed. Objects were represented here at only a single level of scale, but feature descriptions change as a function of observer distance, with larger features dominating at greater distances. The surface data needed to be partly generated by hand, because no surface information was available here. Further, the theory on surface segmentation and description is not well advanced yet. The recognition process is also slow at present, preventing practical application.

The completely recognized robot is significantly more complicated than previously recognized objects (because of its multiple articulated features, curved surfaces, self-occlusion and external occlusion). This success arises because of the ease with which complete, explainable, object recognition can be achieved using surface information and surface-based object models.

1.4 Structure of the Rest of the Thesis

Chapter two presents a critical review of the state of object recognition.

Chapters three through eleven are the body of the thesis. Most of the chapters have a three part structure:

1. motivations and intuitions behind the problem and its solution,

2. the theory and implementation of the proposed solution, and

3. evaluation and critical discussion.

Chapter three deals with the input data requirements. It reviews the sources of surface information, motivates using surface information for object recognition, and considers segmentation of the surface image.

Chapter four examines the question of what is object recognition, how identity is established and proposes the recognition model used in the thesis.

Chapter five covers the model representations as affected by the requirements of recognition. It describes the surface oriented object modeling method and the constraint and association networks used for model invocation.

Chapter six looks at the constraints on reconstructing partially obscured surfaces. It extends classical methods to the richer data in a surface image to overcome occlusion.

Chapter seven introduces an identity-independent object representation called the surface cluster, which groups surfaces to form blob-like solid representations.

Chapter eight presents a variety of 3D properties that can be accurately estimated using 3D surface information (as compared to from a 2D intensity image).

Chapter nine discusses the many factors that play a part in model invocation. A theory that ties many of these together is presented and evaluated. The theory is implementable as a parallel network of simple units, the elements of which are described.

Constructing a complete object hypothesis, as described in chapter ten, requires these actions:

1. estimating the position of the object,

2. deducing what object features should be visible,

3. finding evidence for the model structures, and

4. explaining all data losses caused by occlusion, whether self-inflicted or from external sources.

Chapter eleven introduces hypothesis verification. The key results are surface dependent constraints that help ensure the constructed hypothesis is a physical object, and other constraints that help guarantee object identity.

Chapter twelve concludes the thesis with more extended test results and discussion, including criticisms and suggestions for improvements and extensions, and a summary of the key contributions of the thesis.

Chapter 2

Literature Review

Three dimensional object recognition is still largely limited to blocks world scenes. Only simple, largely polyhedral objects can be fully identified and more complicated objects can only be tentatively recognized (i.e. evidence for only a few features can be found). The research presented in this thesis attempts to bridge the gap.

This chapter examines the current state of object recognition to motivate the use of a surface-based object recognition process.

"Object recognition" is a catch-all category for results on a variety of visual interpretation issues and problems. This chapter reports results relevant to the research presented in this thesis, in the following areas:

1. acquiring and representing surface data for recognition
2. pre-recognition scene understanding
3. object representation for recognition
4. recognition criteria
5. matching algorithms
6. model invocation

7. geometric scene understanding

8. existence and identity verification

The general trends in these areas are discussed in the subsections below.

While this chapter is organized about particular issues in recognition, there are several pieces of research that deserve special mention.

Roberts ([ROB65]) initiated three dimensional model-based scene understanding. Using edge detection methods, he analyzed intensity images of blocks world scenes containing rectangular solids, wedges and prisms. The two key descriptions of a scene were the locations of vertices in its edge description and the configurations of polygons about the vertices. The local polygon topology indexed into the model base, and selected initial model-to-image point correspondences. Using these correspondences, the geometrical relationship between the model, scene and image was computed. A least squares solution accounted for numerical errors. Object scale and distance were resolved by assuming the object rested on a ground plane or on other objects. Recognition of one part of a configuration introduced new edges to help segment and recognize the rest of the configuration.

Marr ([MAR82]) proposed a volumetric model based biological object recognition scheme that:

- took edge data from a $2\frac{1}{2}$D sketch ,
- isolated object regions by identifying occluding contours,
- described sub-elements by their elongation axes, and objects by the local configuration of axes,
- used the configurations to index into and search in a subtype, subcomponent network representing the objects, and
- did geometrical analysis based on image axis positions and constraints from the model.

His proposal was outstanding in the potential scope of recognizable objects, in defining and extracting object independent descriptions directly matchable to 3D models (i.e. elongation axes), in the modeling of subtype and subcomponent model refinement, and in the potential of its invocation process. It suffered from not being evaluated through implementation, from being too serial in its view of recognition, from being limited to only cylinder-like primitives, from not accounting for surface structure and from not fully using the 3D data in the $2\frac{1}{2}$D sketch .

Brooks ([BRO81]), in ACRONYM, implemented a generalized cylinder based recognizer using similar notions. His object representation had both subtype and subcomponent relationships. From its models, ACRONYM derived visible features and relationships, which were then graph-matched to edge image data represented as ribbons (parallel edge groups). ACRONYM deduced object position and model parameters by back constraints in the prediction graph. These symbolically constrained the parameters as a function of the model relationships and image geometry. This symbolic back-constraint and incremental evidence mechanism is superior to the mechanism described in chapter 10, except in two respects: (1) the constraint mechanism does not take explicit account of data errors and so can fail (unless additional bounds on possible error values are added) and (2) the calculation of the symbolic constraints is a significant and imperfect calculation. These problems may be simplified if one uses 3D surface data, rather than data from projected images.

This well developed project demonstrated the utility of explicit geometrical and constraint reasoning, and introduced a computational model for generic identification based on nested sets of constraints. Its weakness was that it only used edge data as input, having a relatively incomplete understanding of the scene, and did not really demonstrate 3D understanding (the main example was an airplane viewed from a great perpendicular height).

Faugeras and his group ([FAU83]) researched 3D object recognition using direct surface data acquired by a laser triangulation process. Their main example was an irregularly cast automobile part. The depth values were segmented into

planar patches using region growing and Hough transform techniques. These data patches were then combinatorially matched to model patches, with the constraint of having a consistent model to data geometrical transformation at each match. The transformation was calculated using several error minimization methods, and consistency is checked by first a fast heuristic check and then by error estimates from the transformation estimation. Their recognition models were directly derived from previous views of the object and record the parameters of the planar surface patches for the object from all views.

Key problems here were the undirected matching, the use of planar patches only, and the relatively incomplete nature of their recognition – pairing of a few patches (it seemed to be 5-6 out of 30) was enough to claim recognition. However, their use of planar patches rather than complete planar surfaces facilitated recognition of a complicated, real cast metal object.

Hanson and Riseman's VISIONS system ([HAN78b]) was proposed as a complete vision system. It was a schema-driven natural scene recognition system acting on edge and multi-spectral region data ([HAN78a]) and used a blackboard system with levels for: vertices, segments, regions, surfaces, volumes, objects and schemata. Various knowledge sources made top-down or bottom-up additions to the blackboard. Identification of objects (road, tree, sky, grass, etc.) used a confidence value on class membership, based on property matching. The properties included: spectral composition, texture, size and 2D shape. Rough geometrical scene analysis estimated the base plane and then object distances knowing rough object sizes. Use of image relations to give rough relative scene ordering was proposed. Besides the properties, schemata were the other major object knowledge source. These organized objects likely to be found together in generic scenes (e.g. a house scene) and provided conditional statistics used to direct the selection of new hypotheses from the blackboard to pursue.

As the system was reported on early in its development, not much evaluation can be made. Its control structure was general and powerful, but its object representations were weak and dependent mainly on a few discriminating properties, with little spatial understanding of 3D scenes.

Bolles et al ([BOL83]) used striper data plus that from a laser range finder. Surface boundaries were found by linking corresponding discontinuities in groups of stripes, and by detecting depth discontinuities in the range data. Matching to models was by using edge and surface data to predict circular and hence cylindrical features, which were then related to the models. The key limitation of these experiments was that only large (usually planar) surfaces could be detected, and so object recognition could depend on only these features. This was sufficient in the limited industrial domains. The main advantages of the surface data was that it was absolute and unambiguous, and that planar (etc) features could be matched directly to other planar features, thus saving on matching combinatorics.

2.1 Sources of Surface Images

Though there are no "perfected" computational processes that produce a surface image, several research areas are leading towards this goal. The major areas are: direct sensing, structured illumination, stereo, optical flow, shading, texture and shape. Each of these processes lead to roughly equivalent information that can produce a surface image. Segmentation and labeling processes then transform the surface image into the representation used by this research. The surface producing processes are surveyed below.

Direct sensing measures the desired properties of the surface directly. Laser ranging is the best example of this. This method computes surface depth by measuring time of flight of a laser pulse or by signal phase shift caused by path length differences. The laser is scanned over the entire scene, producing a depth map that can then be differentiated to give surface orientation (with some difficulties at shape or occlusion boundaries). Sonar range finding gives similar results in air, but has lower resolution and has problems because of surface specularity.

Structured illumination uses controlled stimulation of the environment to produce less ambiguous conditions for interpretation. One well-known technique traces the scene with parallel light stripes ([SHI71], [AGI73], [POP75], [SHN79], [OSH81], [BOL83]). This technique highlights distinct surfaces, because all stripes lying inside the surface boundaries have the same character (e.g. all lines parallel), and usually the character will change radically at occlusion or orientation discontinuity boundaries. Terminations of stripes indicate occlusions. The pattern of stripes on a surface constrains the surface shape, distance and orientation. Planar and cylindrical surfaces are often extracted using this technique.

A second technique uses one or more remotely sensed light spots. By knowing the emitted and received light paths, the object surface can be triangulated, giving a depth map ([KAN81a], [PIP82], [FAU83]). The advantage of using a spot is there is no trouble finding what to correspond in the two images.

Stereo is becoming a more popular technique. It is important because it is a significant biological process ([MAR82], [MAY80]), and because its sensor system is simple and passive. The process is based on finding common features in a pair of images taken at different locations. If the relationship between the camera coordinate frames is known, then the feature's absolute location can be calculated by triangulation. One major difficulty with this technique is finding the common feature in both images. Biological systems are hypothesized to use paired edges with the same sign, and lying in the same spatial frequency range ([MAR82], [MAY80]). Other systems have used detected corners or points where significant intensity changes take place ([DRE81], [MOR81]). Other difficulties arise because stereo is likely to give depth values for only sparse image points, which necessitates surface reconstruction. This topic has only recently entered investigation, but some work has been done using interpolation ([GRI81]), minimal deformation energy of sheet surfaces ([TER83]) and discontinuity merging costs ([BLA84]).

Optical flow arises from the relative motion of the observer and objects, which causes characteristic flow patterns in an intensity image. These flow patterns

can be interpreted to acquire scene distance, surface orientation and occluding boundaries (in the viewer's coordinate system). Horn and Schunck ([HOR81]) did the initial computational work on this problem, showing how spatial variation of the reflectances on the surface of the object related to the time variation of the intensity perceived, as a function of the relative motion. To recreate the surfaces, they assumed local surface smoothness. Nagel ([NAG83]) improved the result by adding an oriented smoothness constraint to account for object boundaries. Reiger and Lawton ([RIE83]) approached this problem directly, detecting object boundaries by looking for local differences in the optical flow field.

These approaches lead to reconstructing the surface depths, from which other information can be derived. Work that has directly estimated these other quantities includes that of Clocksin ([CLO80]) on slant and edge estimation, and Prazdny ([PRA79]) on relative depth.

Shading is a more esoteric source of shape information. It is an obvious physical phenomenon exploited by artists, but has not had much practical success in the reverse process of recognition. Horn ([HOR75]) elaborated the theoretical structure for solving the "shape from shading" problem, and others ([WOO79], [PEN82]) successfully implemented the theory for reasonably simple, uniform surfaces. The method starts from the surface reflectance function that relates reflectance to the illumination, viewer and surface relative orientations. From this, a system of partial differential equations is derived showing how local intensity variation is related to local shape variation. With the addition of boundary, surface continuity and singular point (e.g. highlight) constraints, solutions can be determined for the system of differential equations. The major problem is that the solution relies on a global resolution of constraints, which requires a uniform, characterized reflectance function for the whole surface in question. Unfortunately, few surfaces have a reflectance function that meets this requirement. (Though Pentland ([PEN82]) has shown reasonable success with some natural objects, e.g. a rock and a face.)

Variations of this technique have used multiple light sources ([COL81], [WES82]), polarized light ([KOS79]) or specularities ([BLA85]).

Explicit surface descriptions (e.g. planar, cylindrical) have been obtained by examining iso-intensity contours([TUR74]) and fitting quadratic surfaces ([CER83]) to intensity data.

The shape from shading techniques usually give surface orientation which must then be integrated to give relative, local depth. There is also a problem with the global convex/concave ambiguity of the surface, which arises when only shading information is available. For these reasons, this technique is probably best suited to only qualitative or rough numerical analyses.

Texture gradients are another source of shape information. Assuming texture structure remains constant over the surface, then all variation in either scale ([STE79], [PEN83]) or statistics ([WIT80], [OHT81]) can be ascribed to surface slant distortion. The measure of compression gives local slant and the direction of compression gives local tilt. This information can be used similarly to shading.

The final source of orientation and depth information comes from shape itself. The technique relies on knowledge of how shapes distort with surface orientation, how certain patterns create impressions of three dimensional structure, and what constraints are needed to reconstruct that structure. Examples of this include reconstructing surface orientation from assuming skew symmetry is slant distorted true symmetry ([KAN79]), from maximizing the local ratio of the area to the square of the perimeter ([BRA83]), from families of space curves interpreted as geodesic surface markings ([STE83]), from space curves as locally circular arcs ([BAR83]), and from characteristic distortions in known object surface boundary shapes ([FIS83]). Because this information relies on higher level knowledge of the objects, these final techniques probably would not help the initial stages of analysis much. However, they may provide supporting evidence at the later stages.

Marr's $2\frac{1}{2}$D sketch ([MAR82]) records relative surface depth and orientation, and the types of various boundaries. Another iconic representation that records most of this information and others (e.g. flow fields, reflectance) is the intrinsic image (e.g. [BAR78]).

2.2 Pre-Recognition Scene Understanding

This section considers five topics:

- understanding the 3D structure of the image,
- segmentation of image data into significant units,
- grouping of object-related information,
- description of image data, and
- overcoming occlusion.

Understanding the 3D Structure of the Image

Working primarily in the limited domain of convex polyhedral solids, researchers found that there were enough characteristic phenomena in line drawings of these scenes to isolate and describe the topology and some of the three dimensional structure of the objects without recourse to object models. What was used was knowledge of object properties (e.g. planar surfaces), the scene (e.g. shadows, background and relative surface geometry) and how objects appeared in images (e.g. edge patterns).

Guzman ([GUZ68]) showed that general object and scene heuristics were enough to usually find complete objects and segment them from other objects in the scene.

Huffman ([HUF71]) and Clowes ([CLO71]) formalized and extended the intuitions behind Guzman's reasoning to use the line labels convex, concave, occluding (both sides). Interpreting a full scene involved assigning a consistent labeling to the image boundaries, which eliminated many line configurations that could not correspond to physical objects.

Waltz ([WAL75]) extended the junction and boundary labels for structure caused by shadows, cracks and separable edges and found that adding these increased the likelihood of obtaining a single correct interpretation.

Mackworth ([MAC73]) added new constraints based on quantitative reasoning in a gradient space that eliminated some legally labeled, but unrealizable polyhedra.

Turner ([TUR74]) added labels for curved surfaces, showing how the labels change consistently along boundaries and how illumination properties change across surfaces.

Kanade ([KAN79]) extended the range of scenes and label set for flat laminar surfaces. His scene understanding required topological knowledge from consistent labelings, gradient space constraints and symmetry heuristics.

More recently, other researchers have shown how edges in more general scenes gives 3D shape and placement cues (e.g. [HAN78b], [BIN81], [LOW81]).

Overall, the research made several key points:

- considerable scene analysis could be done using only general knowledge of scene and object phenomena, and without explicit object models,
- these constraints could apply locally to help force a globally consistent interpretation, and
- some of the key features useful for interpreting scenes were: surfaces, boundaries, adjacency and obscuring relationships and shadows.

In this thesis, direct surface data is used, so deducing the label types from image configurations is unnecessary. Sugihara ([SUG79]) used light stripe data to assign labels (concave, convex, obscuring) using a combination of stripe behaviour and valid label configuration rules.

The boundary labels of types occluding (front surface, back surface) and shape discontinuity (convex, concave) are still useful for the reasoning done in

this thesis. Also useful is the understanding of how boundary junctions relate to observer position and vertex structures (e.g. Thorpe and Shafer ([THO83]) as applied to trihedral vertices).

Segmentation of Image Data into Significant Units

Parallel investigations considered grouping image regions. Several researchers ([BRI70], [BAR71]) grouped pixels to form recognition primitives by image intensity subject to merging heuristics. Tenenbaum and Barrow ([TEN77]) improved this by only merging subregions in multispectral images across weak contrast boundaries constrained by (1) having a consistent label set for the two regions and (2) the regions being consistent with other adjacent regions. This work simultaneously segmented and identified the regions, and required label sets based on region interpretations. Several researchers ([HAN78a], [NAG79], [OHT79]) grouped and labeled pixels in outdoor scenes. The need for much domain specific knowledge leaves these segmentation processes questionable except for special purpose analysis. A key problem is that surfaces which produce similar spectral distributions and are adjacent in the image are merged, even though they may not be adjacent in the scene.

Research has proceeded on segmentation and grouping of three dimensional data, which has the advantage of directly corresponding to object surfaces. Various researchers have created surface patch representations from directly sensed data, as in the striper ([SHI71], [POP75], [BOL81], [BOL83]), laser range finding ([OSH81], [FAU83]) or stereo ([POT83], [GRI81]). Bolles et al ([BOL83]) found depth discontinuities in range data and also linked across light stripe junction patterns to find surface edges. This work has not achieved the sophistication yet of the blocks world analysis, and this thesis makes contributions in this area.

An important issue is the appropriateness of the data representation to the vision problem. Typical representations imported from computer graphics are the polygonal surface patch ([BOI81]) and the B-spline ([POT83], [YOR81]). These are geometrical techniques that parametrically represent the surface. Unfortu-

nately, they do not distinguish any of the features needed for recognition (e.g. significant regions, general shape, shape boundaries, relative surface orientation)

Other techniques have concentrated on using natural segmentations, based on the object surfaces themselves. Oshima and Shirai ([OSH81]) and Hebert and Ponce ([HEB82]) make planar or curved surface regions whose boundaries correspond to object surface boundaries. While the regions bounded may not have a simple geometrical shape, the representation is more faithful.

Grouping of Object-related Information

Roberts ([ROB65]) segmented objects by recognising them, which is in the opposite order to our interests. Shirai ([SHI75]) and Waltz ([WAL75]) achieved a rough separation of objects from background by assuming external boundaries of regions were the separator. Heuristics for adding isolated background regions, based on tee matching, were suggested. These techniques required that the background be shadow free, and that the objects did not contact the image boundary.

Both of these approaches concentrated on finding relevant objects by eliminating the irrelevant (i.e. the background). This was later seen to be unprofitable because relevance is usually determined at a higher level. The methods were also incapable of decomposing the object grouping into smaller object groups.

Guzman ([GUZ68]) initiated a sequence of work on surface segmentation using image topology. Starting from line drawings of scenes, he used heuristics based on boundary configurations at junctions to link together image regions to form complete bodies. Huffman ([HUF71]) and Clowes ([CLO71]) put Guzman's heuristics into a more scientific form by isolating distinct bodies at connected concave and obscuring boundaries.

Sugihara ([SUG79]) proposed two heuristics for separating objects in an edge labeled range data image. The first separated objects where two obscuring and two obscured segments meet, depending on a depth gap being detectable from either illumination or viewer effects. The second heuristic separated bodies along

concave boundaries terminating at special types of junctions (mainly involving two obscuring junctions). Other complexities arose because of the disparate illumination and sensor positions.

Neither Waltz (because the labeling would be at best ambiguous) nor Sugihara (heuristics don't apply) could segment a cube lying flush in a corner.

Description of Image Data

The next level of sophistication is in the use of the original data to support the development of intermediate representations, rather than explicitly comprise them. Sloman and Owen ([SLO80]) argued for processing of sketches as impoverished, yet articulated, representations, integrating results from lower processes. The sketches record salient features and provide a basis for decision making on partial information. Hogg ([HOG84]) used edge fragments to provide confirming evidence for generalized cylinder positions.

Several researchers have considered the problems of segmenting surface data into regions useful for recognition. Agin and Binford ([AGI73]) went directly from striper data to generalized cylinder representations. Others ([SHI71], [POP75], [SHN79]) isolated surfaces in striper images by clustering stripes with similar image properties. Fisher ([FIS85a]) proposed that the surface data should be segmented into regions with approximately constant surface curvature and that boundaries should be placed at significant discontinuities in the surface orientation or curvature. The goal of this is to produce surface patches with a constant and characterizable shape. The same criteria was applied to segment 3D boundary curves. Asada and Brady ([ASA84]) discussed similar criteria for segmentation of planar curves. Brady et al ([BRA84a]) investigated determining the surface shape (e.g. lines of curvature) for describing, but not segmenting surfaces.

The most important intermediate representation has been the generalized cylinder. Several researchers have considered how to infer these from a variety of sources. Agin and Binford ([AGI73]) and Nevatia and Binford ([NEV77]) seg-

mented generalized cylinders from range data (from stripers), deriving cylinder axes from stripe midpoints or depth discontinuities.

Marr ([MAR82]) proposed object isolation by occluding contours. These were segmented by convexity properties into elongations described by an axis (i.e. are assumed to be the image projection of a generalized cylinder). This segmentation required distinct protruding or elongated regions, so is only suitable for a limited class of image regions. Stereo data could have been used to provide stronger segmentation criteria (than just using image contours).

Brooks ([BRO81]) described an intensity edge image using ribbons and ellipses, assuming these corresponded to the occluding contour and end pieces of generalized cylinders. The description process was constrained by the expected appearance of the generalized cylinders. Searching for predicted image entities is useful with a small model base, but would fail with a rich model base because of the many primitives seen from many potential viewpoints. The particular image features used here were also relatively limited in interpretive power.

Overcoming Occlusion

Some research has tried to overcome occlusion directly by using visible cues (e.g. ([GUZ68], [ADL75])). The key problem is detection of occlusion, and this work has relied on the use of "tee" detections, which show where one surface boundary is abruptly terminated by the occluding boundary of a closer surface. Because an occluded surface must have a pair of tees at the start and end of the occluding boundary (under normal circumstances), the detection of a matched pair of tees suggests a likely occlusion boundary, and hence where the invisible portion of the surface lies. In the research in this thesis, occlusion boundaries are directly labeled, so the occlusion cueing process is no longer necessary. The tees are still useful for signaling where along the occlusion boundary the occluded surfaces' boundaries terminate. They would also be useful for helping cope with missing, incorrect or ambiguous data (e.g. when a correct boundary label is not available).

2.3 Object Representation for Recognition

Object representations follow two approaches. Property representations define objects by properties or constraints (without recourse to an explicit geometrical model) the satisfaction of which should lead to unique identification. The second representation approach is based around geometric object models. The representations may be expressed implicitly in a computer program or explicitly as a defined model. The implicit case is not different in competence from the explicit, but is ignored here because of its lack of generality.

Marr ([MAR82]) proposed five criteria for object representation:

1. accessibility – needed information in a model should be directly available, rather than derivable through heavy computation,

2. scope – a wide range of objects should be representable,

3. uniqueness – an object should have a unique representation,

4. stability – small variations in an object should not cause large variations in the model, and

5. sensitivity – detailed features should be represented as needed. These criteria will be applied to the techniques reviewed below.

Property Representations

In restricted domains, property representations generally serve as discriminants. Duda and Hart ([DUD70]) used properties like color and height to analyze scenes. Shirai ([SHI78]) used rough sizes, colors and edge shapes to characterize desk top objects. Adler ([ADL75]) used viewer-centered property models to interpret 2D Peanuts cartoon figure scenes. The model primitives were regions with summary properties (e.g. area) while larger figures met adjacency constraints. Falk

([FAL72]) used face shape, edge lengths and 2D edge angles to identify polyhedra. Constraints might also include relationships that have to be held with other structures (e.g. [BAR76]).

Property representations are usually viewer-centered. Minsky ([MIN75]) proposed a frame representation for recording features visible from typical distinct viewpoints. Various researchers ([HAN78b], [NAG79], [OHT79]) have augmented property representations with weak image shape (e.g. parallel, square) and image relations (e.g. above, near).

A more structured property representation is the graph. Here, object features become nodes in the graph and relationships between the features become the arcs. Barrow and Popplestone ([BAR71]) used an viewer-centered graph representing visible object regions and their interrelationships, like adjacency and relative size.

Shneier ([SHN77]) defined a compact relational data structure that merged structure for duplicated or symmetric subcomponents at the cost of loss of detail. The shared representation indexed richer models for more detailed analysis.

Graph representations have the advantage of adding some structure to the object properties, and providing a common representation method for many problems. Uniform domain-independent matching methods can use this general mechanism. One problem is all object details tend to be represented at the same level, so the graphs can become large without benefit. Adding more detail would increase the computational difficulties of matching rather than ease them. Barrow et al ([BAR72]) investigated hierarchical graph representations in matching.

Property representations do well only with Marr's scope criterion. Further, graph and property representations are usually two dimensional, whereas we are interested in three-dimensional objects and scenes, so changes in viewpoint make drastic changes in the representation. Property representations offer simplicity at the expense of having weak descriptive powers and providing no support for active deduction. Further, natural objects are still difficult to represent explicitly

so their recognition must still depend more on special purpose mechanisms or property representations.

Geometrical Representations

Model-based representations embody geometrical models from which views of objects can be deduced, and so support description and active deduction, but at the expense of complexity and substantial computational machinery.

Geometrical models explicitly represent the shape and structure of the object. If the model represents all information needed to describe the appearance of an object, any matchable visual feature could be either directly accessed or derived. Hence, this approach is intrinsically more powerful than the property method. Geometric models often have hierarchical structure, so allow embedding of substructure or refinements. This makes key features prominent, yet leaves other information accessible. Further, for recognition, the models imply geometric constraints on the features (such as the angle between two surfaces) that can be used to help interpret image data. With geometric models, directed matching can take place through the prediction of image feature locations. This helps ease the matching problem.

Point models (e.g. [ROB65]) specify the location of significant points relative to the whole object. This method is simple, but leads to difficulties in correctly establishing model-data correspondences.

Edge models (e.g. [FAL72]) specify the location, orientation and shape of edges (typically orientation discontinuity). These characterize the wire-frame shape of an object better than the point models and have stronger correspondence power, but lead to difficulties because of the ambiguity of scene edges and the difficulty of reliably extracting the edges. Further, when edges are used, curved surfaces have no clear representational device.

Owen ([OWE80]) argued for the use of natural units in representation, and proposed surfaces (as compared to lines) as one such unit for objects. Surface

models describe the shape of observable surface regions and their relationship to the whole object (and perhaps to each other as well).

Wire frame models ([BOL83], [BAL82],pg 291) can also represent the surface regions by their bounding space curves. They most easily represent planar surfaces, but curved surfaces can also be illustrated by judicious placement of lines. While useful for computer-aided design, these tend to omit the surface information needed for recognition, as well as have a uniform level of representation. A good representation for vision needs to have several levels of structural units, to represent both the conceptual units and the hierarchy properly. The wire frames do represent the boundaries between surfaces well, so this feature was adopted, though not in the same manner. Gariboto ([GAR82]) derived 3D axis-based object models from 3D surface descriptions.

Surface patch models give arbitrarily accurate representations of the surface of the object. One approach to surface representation is by bi-cubic spline patches ([YOR81], [BAL82],pg 269), where cubic polynomials interpolate the surface between fixed points, giving both positional and derivative continuity at the points. A second popular approach uses polygonal planar surface patches (e.g. [BOI81]), with splitting of the patches until arbitrary accuracy is achieved. These represent surfaces well, but give no conceptual structure to the surface. For example, one would like to associate labels with particular shapes, or to associate a name with a portion of a whole model (e.g. egg-shaped, or "roof" of an automobile). The surface patches represent at a uniform, un-differentiated level. Also, these modeling approaches ignore the problem of shape (i.e. surface orientation) discontinuities.

Faugeras et al ([FAU83]) used depth data-derived planar patches in a 3D coordinate system to partially bound a 3D rigid object. Here, the model did not characterize the full object, rather it concentrated on significantly planar regions. As their test object was irregular, this approach was useful.

Other researchers have created planar and cylindrical surface models from striper data ([POP75], [DAN82]). Surfaces represent well the actual visibility of an object and allow direct comparison of appearance, but do not easily charac-

terize the mass distribution of an object. Further, criteria for surface description for recognition have not been well formulated yet.

Volumetric models represent the solid components of an object in relation to each other or the whole object. Three dimensional character is directly accessible, but appearance is hard to deduce without the addition of surface shape and reflectance information. Matching with solids requires finding properties of images and solids that are directly comparable, such as occluding boundaries and axes of elongation.

Space filling models ([BAL82],pg 280) represent objects by denoting the portions of space in which the object is located. This representation meets only Marr's scope and uniqueness criteria. storage, and does not explicitly differentiate between visible (surface) and invisible (interior) portions of the model.

Constructive solid geometry starts from geometrical primitives like cubes, cylinders or half-spaces ([REQ77], [CAM84]) and then forms more complex objects by merging, difference and intersection operations. The primitives and modeling operations are simple, but, unfortunately, this approach makes little of the required information explicit. One particular requirement is to identify what bits of the model primitives lie on the object surface. The notion of making more complex objects from smaller units is important though, and the recognition models used in this research apply this concept, though as a union of disjoint solids.

The most promising models proposed so far are the generalized cone or cylinder models ([BIN71], [AGI79], [MAR78], [HOG84]), which have had their most significant usage in the ACRONYM system ([BRO81]). These models are structured, so meet the sensitivity criterion and give unique representations. When modeling objects formed from hierarchical collections of elongated primitive shapes, the generalized cylinder method also meets the scope and stability criteria.

The primitive unit of representation is a solid specified by a cross sectional shape, a sweeping rule and an axis along which to sweep the cross section. The

shape and angle to the axis of the cross section can vary as a function of the position along the axis. The axis was the key feature because of its relation to axes directly derivable from image data. Many "growth" based natural structures (e.g. tree branches, human limbs) have an axis of elongation, so generalized cylinders make good models. It also represents many simple man-made objects well, because the manufacturing operations of extrusion, shaping and turning create reasonably regular, nearly symmetric elongated solids.

In Marr's proposal ([MAR82]), structures were described by the dominant model axis and the names and distribution of subcomponents about the axis at each level in the model. Subcomponents could be refined to provide greater detail. Subcomponents were placed by the six degree of freedom relationship between their axes and the main axis. The specification used dimensionless units, which allowed size invariance, and the relative values were represented by quantized value ranges that provided both the symbolic and approximate representation needed for stability to variation and error. This model scheme is impressive in its attention to representing information directly relatable to image properties in symbolic form and its subcomponent refinement.

Brooks ([BRO81]) used generalized cylinders as volumetric model primitives because they could be directly matched to image boundary pairs (i.e. ribbons) and also represented many elongated shapes well. More complex structures (e.g. airplanes) were formed by aggregating subparts. These subparts were attached by specifying the rotation and translation relating the object and subobject reference frames, which was not as conceptually "nice" as Marr's proposal where the affixments were specified using elongation axis relationships. All primitives or affixment relationships could contain variables. Inequality constraints on these variables then structured the space of all possible models with the given logical part relationships into a generalization hierarchy, where more restrictive constraints define generic specializations of the model.

The subcomponent generalized cylinders in ACRONYM's airplane models were all rigidly connected. Hogg ([HOG84]) used variable attachments to rep-

resent joint variation in a human model, with a posture function constraining relative joint position over time.

For use, ACRONYM's geometrical models were compiled into the prediction graph. Here, key generalised cylinders became the nodes and placement relationships between the cylinders defined the relations. Because the image data being matched was 2D, the prediction graphs represented typical 2D views of the objects. These were derived from the full 3D geometrical model. The advantage of this was the full constraints of the geometrical model could be employed in the uniform graph matching method. Substantial reasoning is needed to derive the prediction graph from the 3D models. (This is still an open research area, e.g. [MAY85].)

The final feature, and most important contribution, of ACRONYM's modeling is the use of constraints. Constraints limit the range of a variable, in relation to either fixed values or other variables. An example would be: "the length of a desk top is greater than the width, but less than twice the width", where both length and width are variable parameters.

Variables and constraints together support generic class models, at least in structural terms (as compared to functional). The structural aspects of the model define the essential components and their attachments, symbolic parameters denote the type of variation and the constraints limit the acceptable range of variation among the members of the class. Small classes have tightly constrained (or constant) parameters. Subclasses may have additional constraints added.

The reduction of generics to numerical ranges of parameter values, while an important first step, is simplistic. Sometimes it is inappropriate: a model adequate for recognising a particular type of office chair probably would not specialise to any other chair, nor would any relaxing of parameters be likely to include many other types. Relaxing the constraints sufficiently to include most office chairs would require replacing the structural notions of a chair with functional notions: seating surface meets appropriate back support surface. The physical variety of both natural and man-made objects is not well suited to generalisation structuring by scale alone.

Brooks attempted to introduce structural variation through parameter variation, but the solutions seem inappropriate. For example, a integer variable ranging from 3 to 6 was used to state that an electric motor had 3,4,5 or 6 flanges, and a second variable stated that a motor did or did not have a base by constraining its value to 0 to 1. More complicated algebraic inequalities stated that motors with bases have no flanges. Uniform representation is a laudable goal, but these examples suggest that a more powerful representation should be considered.

In summary, the physical variation within a class, which constraints represent well, should be separated from the conceptual relationships involved in generalization. That is, each object and object class should have its own models and sets of constraints, but there should not be a strict subset relationship between the subclass and class. New, incompatible constraints should be allowed to introduce variation and exceptions to the generalization. Secondly, the different types of constraints should have different representations. Numerical ranges are suitable for size and affixment variations, but logical/relational constraints would be better for subclass representation.

Another criticism of the generalized cylinder/cone representation is on its choice of primitive element. Many natural and man-made objects do not have vaguely cylindrical components: a leaf, a rock, the moon, a crumpled newspaper, a tennis shoe. Though some aspects of these objects could be reasonably represented, the representation would omit some relevant aspects (e.g. the essential two dimensionality of the leaf), or introduce other irrelevant ones (e.g. the axis of a sphere). Hence, other primitives should at least be included to increase its descriptive adequacy.

Secondly, what is perceived is the surface of objects, hence it seems reasonable that the preferential representation for object recognition should make surface-based information explicit. The near-parallel, tangential occluding boundaries of a generalized cone ("ribbon") are reasonable features for detection, and the orientation of the figure's spine constrains its 3D location, but this is about all the information easily derivable from the cone representation. Surface shape

comparisons are non-trivial, because it is difficult to determine the visible surfaces of the cone and what a cone will look like from a given viewpoint. It is often hard to decide with what model features a piece of image evidence should correspond.

Using a direct surface representation, one can easily predict the location of shape and occluding boundaries and the surface's perceived shape. The local character of any surface region is immediately determinable. Because there is often a one-to-one correspondence between the model and surface image boundaries, it is possible to estimate the geometric transformation from model to scene ([FIS83]).

These considerations, however, are most relevant when recognizing objects whose surface shapes and structures are apparent at the scale of analysis. The ACRONYM examples, aerial photographs of airport scenes, are largely 2D as almost all objects were reduced to laminar surfaces viewed perpendicularly. Hence, treating nearly parallel intensity boundaries as potential occluding boundaries of the projection of generalized cones was appropriate.

Another volumetric representation is that proposed by Shapiro et al ([SHA80]). This gave a rough 3D object model based on sticks (1D), plates (2D) and blobs (3D) and a characterization of their structural relationships. It was intended for use in a relational matcher.

Links to the Modeling Used in this Thesis

The representations used in this thesis have much in common with previous visual representation systems. The use of descriptive attributes of objects for discrimination has been the approach of pattern recognition or discrimination net identifiers, or for scene analysis in restricted domains ([TEN73], [TEN74], [SHI78]). In this thesis, descriptions are not used for identification or verification, but for suggestive invocation – identification comes from comparison with models. Relational descriptions of structure and form of objects can be created. In the thesis, the model graphs represent object-based information, rather than

image-based information, and the relationships are used to suggest models, or to confirm hypotheses, but not to create hypotheses.

Object-centered representations with geometrical affixments ([BRO81]) and hierarchical structure have been used successfully before and are used prominently in the geometrical object representation.

2.4 Recognition Criteria

The criteria for declaring an object recognized fall into four categories. These are summarized below roughly in order of discriminating power. Most recognition systems use criteria from several.

Sufficient properties

When enough model properties are satisfied by the data, recognition is declared. The properties may be scene location, orientation, size, color, shape or others. The goal is unique discrimination in the model base, so judicious choice of properties is necessary. Duda and Hart's ([DUD70]) analysis of an office scene typified this. Brice and Fennema ([BRI70]) classified regions by their boundary shape and object identity was defined by a group of regions with the correct shapes. Adler ([ADL75]) ranked matches by success at finding components meeting structural relationships and summary properties (for primitives). Falk ([FAL72]) and Bolles et al ([BOL83]) also followed this, except their choice of structural properties gave stronger matching. Property comparison is simple and efficient, but is not generally powerful enough for a large model base or subtle discriminations. The problem is always – "what properties?".

Given that properties are not exact, a distance metric is often used (e.g. [TUR74]) to evaluate the match (a typical pattern recognition method).

Grammatical, graph or template matching

When a particular grouping of data is identical to a similar model pattern recognition is achieved. This usually expresses relationships between image features, such as edges or regions, but may also refer to relationships in 3D data. This method requires evaluation of the match between individual features. Rosenfeld ([ROS72]) presented a typical example of this matching in his web grammars for analysing 2D patterns. Barrow and Popplestone ([BAR71]) used a heuristic weighting to evaluate the satisfaction of a subgraph match, including a factor that favored larger matches. Ambler et al ([AMB75]) used similarity of properties and relations between regions in a 2D parts scene. Nevatia and Binford ([NEV77]) evaluated matches based on components found and parameter comparisons for the primitives. Hogg ([HOG84]) used predictions from image tracking to dynamically create templates for feature verification, and required sufficient oriented edge points inside these templates. The templates were produced by projection from a positioned 3D generalized cylinder model.

This approach has the advantage of easy computation through symbol matching, formal definition and computationally analysable machinery. One disadvantage is that 3D scenes have changing viewpoint and occlusion, which distorts and fragments object descriptions (unless multiple graphs are used for alternative viewpoints).

Geometrical matching

Recognition is the satisfaction of geometrical criteria – such as the accumulated error between predicted and observed features being below a threshold. Roberts ([ROB65]) used model – data approved polygon topology matching subject to a least square error threshold on reference frame estimation. Faugeras and Hebert ([FAU83]) used the data-to-model surface pairings that passed a geometrical consistency measure and had minimum transformation estimation error. Fisher ([FIS83]) declared an object recognized if all paired model-data structures had the correct reference frame relationship.

Correct geometry provides a strong constraint on an object's identity. Its limitations include the requirement for pairing the appropriate structures, control of combinatorial matching and integration with other matchables: structure properties and relationships.

Constraint satisfaction

Implicit in the above methods are constraints that the data must satisfy. Some researchers have tried to generalise this by making the constraints explicit. Hinton ([HIN76]) and Paul ([PAU76]) refined object identity from constraints of possible component identities and component proximity (according to the model). These results were applied to 2D puppet models. Barrow and Tenenbaum ([BAR76]) used adjacency and homogeneity constraints to deduce identity in office scenes using height, intensity and orientation data. Marr ([MAR82]) argued that recognition was the refinement of the specificity of description in a generic hierarchy, but did not propose specific matching or acceptability criteria. In ACRONYM ([BRO81]), an object was identified by maximal refinement in a specialization lattice consistent with both the model constraints and image data. The refinement was by constraint satisfaction, where the constraints largely covered feature sizes and relationships.

One important problem with constraint satisfaction is how to cope with the occasional constraint violation due to noise. (This is also a problem with the other three criteria types as well.) Allowing a few violations would prevent most failures, at the high cost of discrimination power, as many objects differ in only a few attributes. Using an error measure to arbitrate leads to problems with deciding when a match is close enough. If the constraints were weakened slightly to tolerate erroneous results, but many constraints were employed, then dissimilar objects would remain distinguished and similar objects would not be misidentified unless their distinguishing characteristics were similar as well.

The best approach is this satisfaction of constraints, because it potentially encompasses the other methods. The ability of a constraint to be general is a

real advantage, particularly when representing ranges of numerical values. Its weakness is it requires the choice of constraints that efficiently and adequately discriminate without rejection of minor undistinguished variants. In particular, with constraint satisfaction:

- matching should occur between symbolic entities,
- individual properties of these entities should meet constraints, and
- relationships (especially geometrical) between entities should meet constraints.

2.5 Matching Algorithms

This section considers how to achieve the match criteria.

Property Matching Algorithms

Simple property matchers use discrimination methods, implemented as sequential property comparison, decision trees or distance based classifiers. These are straightforward, but do not easily allow complicated recognition criteria (e.g. geometrical or relational) without prior calculation of all potential properties, and treat objects at a single level of representation.

Syntactic and Graph Matching Algorithms

For objects with primitive distinguishable features having fixed relationships (geometrical or topological), two general methods have been developed. The first is the syntactic method (e.g. [MIL68], [ROS72], [CHA79]). Valid relationships are embodied in grammar rules and recognition is by parsing the data symbols according to these rules. Their primary use has been in fingerprint ([MOA76]), circuit diagram, chromosome and texture analysis. A variation on

this method uses rules to recognize specific features (e.g. vegetation in an aerial image ([NAG79]) or urban building scene ([OHT79])).

The second general technique is graph matching, where the goal is to find a pairing between subsets of the data and model graphs. The two key techniques are subgraph isomorphism and maximal clique finding in association graphs ([BAR74]). Barrow and Popplestone ([BAR71]) used a subgraph matching between their data and model graphs. Ambler et al ([AMB75]) recognized by a maximal clique method in an association graph between data and models. Combinatorial explosion was controlled by using a hierarchy of structures ([BAR72]). Turner ([TUR74]) exploited this method procedurally in his hierarchical synthesis ([SEL60]) matcher.

The advantage of graph matching is that it is well understood. The disadvantage is that these methods tend to be NP complete algorithms and are not practical unless graph size is small. Matching would be more efficient if geometrical predictions were used, allowing direct comparison instead of the complete matching that general graph matching algorithms require. Finally, heuristic match criteria are still needed for comparing nodes and arcs, and for ranking subgraph matches.

Constraint Satisfaction Algorithms

A third group of general algorithms are those for managing constraint satisfaction criteria. The algorithms can use direct search, graph matching where the constraints specify the node and arc match criteria, or a parallel relaxation algorithm. Relaxation algorithms can apply to discrete symbol labelings ([WAL75]), probabilistic labelings ([ZUC77], [ROS78], [BER83], [FAU80]) or a combination of the two ([BAR76]). Hinton ([HIN76]) formulated the substructure identity problem as a relaxation problem, with the goal of maximizing credibility subject to model constraints. Nevatia and Binford ([NEV77]) matched models using connectivity relationships between generalized cylinder primitives in the model and data to constrain correspondences. Brooks ([BRO81]) further developed

techniques that reduce sets of algebraic constraints. This was to determine if sets of constraints were consistent or to estimate parameter values.

The goal of most algorithms was to use local constraints to produce global consistency. The difficulty with these pure methods is that they simplify excessively and ignore most of the global structural relationships between nameable object features.

Geometrical Matching Algorithms

The final matching method is geometrical. Here, the geometrical relationships in the model, initial object location knowledge and image feature geometry combine to allow direct matching. Roberts ([ROB65]), Freuder ([FRE77]), Marr ([MAR82]) and others argued that partial matching of image data to object models could be used to constrain where other features were and how to classify them. Locating this data then further constrained the object's geometrical location and as well as increasingly confirmed its identity. Adler ([ADL75]) used a top-down control regime to predict image location in 2D scenes, with demons to explain data loss because of occlusion.

Freuder ([FRE77]) described a 2D recognition program that used active reasoning to recognize a hammer in image region data. The program used image models to obtain suggestions of what features to look for next and advice on where to find the features.

Matching may be almost totally a matter of satisfying geometrical criteria. The advantage of geometrical matching is that matching criteria are usually clear and geometrical models allow directed comparisons. Roberts ([ROB65]) initiated geometrical matching by solving for the transformation that mapped selected model points to image points. Mapping errors exceeding a threshold implied a bad match. Ikeuchi ([IKE81]) recognised and oriented objects by computationally rotating extended gaussian images until good correspondences were achieved. Hogg ([HOG84]) improved positional estimates using search over a bounded parameter space. Ballard and Tanaka ([BAL85]) used a connectionist

method for deducing a polyhedral object's reference frame given network linkages specified by geometrical constraints. This follows Ballard's work ([BAL81a]) on extracting component parameters from intrinsic images using Hough transform techniques.

Several systems used a combination of the methods to recognize objects in more sophisticated scenes. ACRONYM's ([BRO81]) matching algorithm looked for subgraph isomorphisms between the picture graph, representing located image features, and the prediction graph, which was a precompilation of the object models. This graph tried to represent the likely sizes and intercomponent relationships between primitives, as seen in typical views of the object. A node or arc match required not only local satisfaction of predicted constraints, but also satisfaction of global constraints such as all features potentially having the same reference frame. Barrow and Tenenbaum ([BAR76]) used best first search with a relaxation based evaluation process. Faugeras and Hebert ([FAU83]) used full combinatorial matching between model and data surfaces, subject to geometrical transformation constraints. Bolles et al ([BOL83]) matched surfaces by property, such as curvature and dimension, and objects were found by aggregating features in pairs consistent with the model.

An improvement on the above basic methods is hierarchical recognition, in which objects are structured into levels of representation, and recognition matches components at the same level. Turner ([TUR74]), Ambler et al ([AMB75]) and Fisher ([FIS83]) used a bottom-up "hierarchical synthesis" process and Adler ([ADL75]) used top-down model directed analysis.

2.6 Model Invocation

To date, little work has been done on sophisticated model invocation in the context of 3D vision. The most common technique is comparing all models to the data. This is useful when only a single item (e.g. industrial part) is desired or when there are only a few possibilities (e.g. parts coming down an assembly line).

A second level uses a few easily measured object (or image) properties to directly select a subset of potential models for complete matching. Roberts ([ROB65]) used configurations of approved polygons in the line image to directly index models according to viewpoint. Nevatia and Binford ([NEV77]) used an indexing scheme that compared the number of generalized cylinders connecting at the two ends of a distinguished cylinder.

Key properties are clearly needed for this task, so this was a good advance. Its limitations (at this stage) are not considering more general classes of evidence, including object relationships; being sensitive to property estimation errors and being too monolithic. Recognition should be incremental and share common subfeature recognition. Property-based indexing either makes subfeatures unusable (properties too complex to calculate everywhere or too object specific), or invokes everywhere (properties too simple and common) or does not properly account for commonality of substructures. Shneier ([SHN79]) proposed a compact relational scheme that shared features common to several models.

There is little AI or vision research that treats model invocation as a specific issue. The general AI research has tended to focus on meta-level rule invocation, which is only remotely related. Work by Schank and associates (e.g. [SCH75]) has focused more on the contents and use of models (schemas) and has not reported how a schema is selected. Minsky ([MIN75]) claimed that a frame must be selected to organise the image data, but avoided the problem of initial frame selection. He suggested how alternative frames can be selected by detecting shortcomings or a need for elaboration. This concept was evaluated

in the use of prototypes for organising knowledge in the CENTAUR expert system ([AIK79]). This approach embodies too much discrete control for the initial levels of invocation. What is needed is a flow of plausibility, not control.

Visual Model Invocation

Roberts ([ROB65]) initiated the visual invocation problem by using the topology of polygons at an image vertex as the index. This was an ideal solution for his limited object domain, but it is not adequate for more realistic domains.

Marr stated:

> "Recognition involves two things: a collection of stored 3-D model descriptions, and various indexes into the collection that allow a newly derived description to be associated with a description in the collection." ([MAR82], pg 318)

While invocation is needed for more than just 3-D models, the general principle seems sound. He advocated a structured object model base linked and indexed on three types of links: the specificity, adjunct and parent indices. (These correspond to the subtype, subcomponent and supercomponent link types proposed in this thesis.) He assumed that the image structures are well described and that model invocation is based on searching the model base using constraints on the relative sizes, shapes and orientations of the object axes. Recognized structures lead to new possibilities by following the indices.

Direct indexing will work for the highest levels of invocation, and with perfect data from perfectly formed objects, but it is probably inadequate for more realistic situations. Further, he avoided the problem of locating where to start the search from, particularly in a large model base. This view of invocation is more like detailed classification, once generic recognition has taken place.

His representation recorded much of the information appropriate to invocation: key properties and interobject relationships in a generic and subcomponent

hierarchy. However, it neglected problems of suggestivity, would probably fail under incomplete evidence, was not incremental and was too serial in outlook.

The ACRONYM system ([BRO81]) implemented a similar notion.

Arbib ([ARB79]) proposed a schema-based invocation process that is similar to that proposed and implemented here. He argued that invocation takes place in a schematic context. Schemas have three components:

> "i. Input-matching routines which test for evidence that that which the schema represents is indeed present in the environment.
>
> ii. Action routines – whose parameters may be tuned by parameter-fitting in the input-matching routines.
>
> iii. Competition and cooperation routines which, for example, use context (activation levels and spatial relations of other schemas) to lower or raise the schema's activation level."

Instances of schemas are invoked to explain image data. His point (i) requires each schema to be an active matching process, rather than the evidence accumulation process proposed in chapter 9. However, his proposal is similar to the direct evidence plausibility computation. His point (ii) corresponds to the hypothesis completion and verification processes (chapters 10 and 11) and point (iii) corresponds closely to the inhibition and association evidence type (chapter 9). He suggested an inhibition association based on exclusive interpretation of evidence. This is probably valid for recognition (i.e. we "see" only one interpretation at a time), but not for invocation as all reasonable interpretations must be ready for consideration. The schema invocation process is not formally defined, nor is there any notion of the invocation ordering. Further, his invocation discussion only focuses on the highest levels of description (e.g. objects) and only weakly on the types of visual evidence or the actual invocation computation.

Bolles et al ([BOL80]) implemented a powerful method for practical indexing, in their local feature focus method (for use in a 2D silhouette industrial domain). The method used key features (e.g. holes and corners) as the primary indices

(focus feature), which were then supported by locating secondary features at given distances from the first. This also oriented the parts.

There are also several object recognition processes that discriminate among a small set of models using observed evidence. Examples include the early SRI work (e.g. [BAR76], [TEN73], [TEN74]) recognizing objects in office scenes using constraints that hold between objects. Object properties are used to discriminate between potential objects in the domain using tabular and decision-tree techniques. This work confused recognition with invocation, but, because the model bases were small, the domain simple and the objects simply discriminable, the technique worked. If model bases are large, then there are likely to be many common properties held by each object, so unique discrimination will be hard. Further, data errors, generic objects and occlusion (by self or other objects) will make the choice of initial index property difficult, or require vast duplication of index links. The general problem requirements make this solution complex.

Hinton proposed ([HIN81]) and evaluated ([HIN85]) a connectionist (see below) model of invocation that assigns a reference frame as well as invokes the model. The model proposed connections between retinotopic feature units, orientation mapping units, object feature (subcomponent) units and object units. This model required duplicated connections for each visible orientation, but expressed these through a uniform mapping method. Consistent patterns of activity between the model and data features reinforced the activation of the mapping and model units. The model was proposed only for 2D patterns (letters) and required many heuristics for weight selection and convergence. Further, both direct (data feature) evidence and indirect (model feature) evidence was used identically, only with different weights.

Feldman and Ballard ([FEL83]) proposed a connectionist model indexing scheme using spatial coherence (coincidence) of properties to gate integration of evidence. This helps overcome invocation due to coincidentally related features in separate parts of the image. The properties used in their example are simple discriminators: "circle, baby-blue and fairly-fast" for a frisbee. This complements Marr's proposals, in that it is parallel, integrates a variety of properties

and need not require perfect satisfaction of a conjunction. The shortcomings here are in not having a rich representation of the types of knowledge useful for invocation. Their proposal did not carefully question what types of evidence are integrated, but proposed a detailed computational model for the elements and their connections.

Feldman ([FEL85]) later refined this model. It starts with spatially co-located conjunctions of pairs of properties connected in parallel with the feature plane (descriptions of image properties). Complete objects are activated for the whole image based on conjunctions of activations of these spatially coincident pairs. The advantage of complete image activation is that then it is not necessary to connect new objects in each image location. The disadvantage is in increased likelihood of spurious invocations arising from cross-talk (i.e. unrelated, spatially separated features invoking the model). Top-down priming of the model holds when other knowledge (world) is available. Structured objects are represented by linkage to the subcomponents in the distinct object viewpoints. Multiple instances of the objects use "instance" nodes, but little information is given to suggest how the whole image model can activate separate instances.

This proposal is similar to the results in chapter 9 in its general character: direct property evidence triggers structurally decomposed objects seen from given viewpoints. Feldman considered the problem of implementation in a massively parallel machine more carefully (than in chapter 9), but did not consider generic evidence, nor the precise nature of the computation implemented in the connections.

General AI Invocation Methods

The NETL formalism of Fahlman ([FAH80], [FAH81]) is a general indexing approach to invocation. This approach creates a large net-like database, with generalization/specialization type links. One function of this structure is to allow fast parallel search for concepts based on intersections of properties. For example, an elephant node is invoked by intersection of the large, grey and mammal

properties. The accessing is by way of passing markers about the network (implemented in parallel). The discrete formulation with few links currently makes it difficult to implement suggestiveness, as all propagated values must be based on definite (i.e. certain) properties. Strength of evidence and specific link types for visual invocation would be needed extensions to this work.

Other Potential Techniques

The possibility calculus of fuzzy logic ([ZAD79]) is similar to the plausibility computation described in section 9.2. Fuzzy logic is a mechanism for approximate reasoning offering a schema for translation of natural language statements to functions over sets, and a generalized modus ponens inference mechanism. It is based on associating a set (e.g. BIG) with a concept (e.g. "big"), whose entries record the possibility that members of the domain have that attribute, given various properties (e.g. size).

We would like to integrate information of the form: "a bright, disk-like object seen in the sky might be the sun". Fuzzy logic might evaluate the possibility of being the sun as:

$$\min\left(\mu_{\text{disk}}(X), \mu_{\text{bright}}(X), \mu_{\text{in\_sky}}(X)\right)$$

where the μ's are appropriate possibility functions. Hence, the fuzzy logic formulation would imply that the object could not be any more possible than the worst of its evidence. This approach does not integrate evidence well, so having two of the three properties strongly held does not improve the resulting possibility. Another problem is that fuzzy logic is a general mechanism, whereas we will want reasoning specifically tailored for invocation computation, which has both direct and circumstantial evidence. Finally, we would like more of a three-valued logic flavor: contradictory, unknown, confirming, with contradictory evidence having stronger weight than confirming evidence (as other objects may have the same confirming property). Hence, fuzzy logic does not seem entirely appropriate for use here.

General pattern recognition/classification techniques are also of some use in suggesting potential models. A multi-variate classifier (see [DUD73]) could be used to assign initial direct evidence plausibility to structures based on observed evidence. Unfortunately, this mechanism is good at properties, but not at integrating evidence from other sources, such as from subcomponent or generic relationships. Further, it is hard to provide the a priori appearance and property statistics needed for the better classifiers.

A computational mechanism that is receiving increasing AI interest is the connectionist computation, in which the domain knowledge is expressed as the interobject relationships. These are made explicit in the weighted interconnections between simple processing units. Hopfield ([HOP84]) has shown how such a machine can converge to a fixed output state for a given input state, based on a minimum energy paradigm. The interpretation of this effect is that the output represents the desired computation on a given input, yet no explicit *algorithmic* description of the process is required. Hinton, Sejnowski and Ackley ([HIN83],[ACK85]) have proposed a variant (the Boltzman machine) that can learn network connection weights, and converge by a simulated annealing process. These machines are currently proposed as linear binary processors, but they should be easily generalizable. Minsky and Papert ([MIN69]) thoroughly investigated the properties of a simple computational device, the linear threshold element, when used in large (e.g. parallel) groupings without feedback and proved several results. Many of the computations proposed for invocation (chapter 9) can be implemented using such devices and their limitations do not cause significant problems.

A difficulty with this research from the viewpoint of visual invocation is that they are mechanisms without problems. In chapter 9, a network formulation for invocation is proposed whose parallelism is useful for two reasons: (1) the need for fast retrieval and (2) the network structure is a convenient formalism for expressing the computational relationships between evidence types. Though the theory proposes direct information channels between processes, these processes

could be implicit in a connectionist network; only the computational structure is outlined, not the implementation.

The relaxation-based vision processes are also peripherally relevant, because the plausibility refinement computation is similar to this class of computations. Each image structure has a set of possible labels that must be consistent with the input data and related structure labels. Applications have tended to use the process for either image modification ([ROS78]), pixel classification ([HAN78a]), structure detection, or discrete consistency maintenance ([WAL75]). Most of the applications modify the input data to force interpretations that are consistent with some criterion rather than to suggest interpretations that are verified in another manner. Invocation must allow multiple labels (generics) and has a non-linear and non-probabilistic formulation, that makes it difficult to apply previous results about relaxation computations.

In summary, there has been some important work leading to the invocation process described in chapter 9. Unfortunately, the work has tended to be either fragmentary, directed at other problems, not used much object knowledge, considered only for simple domains, or was only a proposal.

2.7 Geometrical Scene Understanding

There are several levels of geometrical scene understanding in the research surveyed. These are summarized in the following points.

A. Pattern Recognition Techniques

These techniques often allow one to say roughly where the object is in the image, but do not provide precise placement, description and feature correspondences.

B. Geometrical Image Understanding

B1. Topological Correspondences between Image and Model Features

Graph matching (e.g. [BAR71]) typifies this level, which allows correspondences between image and model features, but not scene placement nor precise image description.

B2. Image Level Spatial Relations (e.g. above, left, near)

Here, rough spatial relations are found between image features and an image model (e.g. [HAN78b], [NAG79], [OHT79], [ABE83]). This also includes environmental relations like the sky being at the top of a image and above roofs ([OHT79]), or most scene lines being vertical ([KEN83]). These allow for rough correspondences and object placement in the image.

B3. Image Level Geometry

Geometrical parts models and a known camera-scene relationship allow deduction of 2D translation and rotation parameters, precise image prediction and precise image feature correspondence and labeling.

Rigid 2D objects and scenes are simpler, as appearance geometry is constant up to 2D translation and rotation, and it is often possible to find all object features (barring occlusion loss). With a geometrical object model, knowing the object's position makes it possible to predict where image features lie (e.g. [BOL80], [LUX83]). Two dimensional hypothesis verification processes often use this method. Determining the object position and orientation is easy, because the relative position of object features is constant and the object-image relationship is simple.

Occlusion understanding in 2D (e.g. overlapping parts) has been less successful until recently, because data loss has made the feature pairing more difficult. Fortunately, only a few local features are needed to orient the whole object, which

allows prediction of other feature locations, as several researchers (e.g. [PER77], [BOL80], [YIN81]) have shown in 2D. Bhanu ([BHA83]) used a stochastic labeling process to segment and identify obscured object silhouettes, with constraints on unique interpretation and null class identifications for obscured segments.

C. Images to 3D Scene Understanding

C1. 3D Location and Partial Appearance Understanding

The most important results were discussed at the beginning of the chapter (Roberts, Marr, Brooks).

Typical scene understanders used point (e.g. [ROB65], [FAL72]), corner or edge correspondences compared to geometrical models to solve for object location and projection relationships. Object 3D location can then be used to predict verifiable image features. Turner ([TUR74]) located objects in 3D using stereo triangulation on identified feature points. Ballard and Sabbah ([BAL81b]) used a variety of Hough transformation techniques to sequentially estimate the 6 positional parameters. This uniform mechanism was more stable to noise, but is likely to suffer when object's shape varies dramatically with viewpoint. It did not depend on any real understanding of the object shape. For two dimensional structures, Brady and Yuille ([BRA83]), Kender and Kanade ([KEN80]), Barrow and Tenenbaum ([BAR80]) and Kanade ([KAN81b]) have considered problems of estimating the spatial orientation from symmetry and parallelism properties. Fisher ([FIS83]) used boundaries to estimate model surface placement, and then used the surface relations to estimate object positions.

Turner ([TUR74]) attempted a more symbolically descriptive matching by classifying surfaces region shapes by the patterns of iso-intensity curves. The elementary recognition operation was by property and relation matching. More complicated objects were recognized by aggregating sub-objects in a hierarchical synthesis process, which is heavily used in the results discussed in chapter 10. The use of more descriptive surface regions and a structured recognition process

were the key contributions of this research, and the key limitations were the simplicity of objects recognizable through using only relational models.

Research in this area has been limited to complete image understanding of simple geometrical objects (e.g. [ROB65]) or partial understanding of complex assemblies of simple objects ([BRO81]). Irregular objects are not well understood at this level, in part because of problems with object modeling and in part because of the difficulty obtaining useful image data.

C2. Full Object Appearance Understanding

(predicted appearance, occlusion explanations and all features accounted for)

This level of understanding has only been achieved for blocks world scenes.

Several researchers have used object models and 3D object location information to predict the location and appearance of image features. Falk ([FAL72]) predicted lines in a blocks world domain, and Freuder ([FRE77]) predicted image region locations in 2D using procedural models of hammers. More recently, Brooks ([BRO81]) showed how a range of image positions could be predicted using partial constraints on object location.

Hogg ([HOG84]) used edge point information to verify the positional parameters of a generalized cylinder human model over time in a natural scene. Individual evidence was weak, but requiring evidence for the whole complex model led to good results.

Some structural understanding of shadows in 3D scenes has been achieved ([NAG79], [LOW81]).

Occlusion in three dimensions has had few results to date. Blocks world scenes were usually successfully analyzed by Guzman's heuristics ([GUZ67]). These included the "paired tee" occlusion identification and image region pairing heuristics. Brooks ([BRO81]) suggested that an intelligent vision system with object models and image geometry understanding could predict self-obscured features (as well as which features would be visible from particular viewpoints).

The occlusion results in chapter 10 are derived from this suggestion, with the addition of image feature reasoning for verifying the description of obscuring structures, needed for occlusion caused by unrelated structures.

Koenderink and van Doorn ([KOE82]) characterized occlusion by their local surface relationships, and showed how the occlusion signatures progressively vary as viewpoints change. This micro-level occlusion understanding was not used in this research, but would be useful for predicting local surface shape for verification of hypothesized occlusion.

D. Surface Data to 3D Scene Understanding

This is like C above, only using surface data rather than image intensity data. Surface data simplifies the geometrical calculations because (1) spatial position is more completely constrained and (2) image feature interpretation is less ambiguous. No work has attempted the equivalent of C2 for surface data yet, but there is some work at the simpler level. The best results are by Faugeras and Bolles (as discussed above).

Surfaces have been used for recognition since the early 1970's. Several researchers (e.g. [SHI71], [POP75]) collected surface data using a structured light system, where configurations of light stripes were used to characterize regular surface shapes. This method of data collection has again become popular (e.g. [OSH81]).

Ikeuchi's ([IKE81]) extended gaussian image method reduced object description to a sphere with quills representing the sizes of areas with the corresponding orientations. This approach matched 3D data, but ignored structural features of the object, and seems likely to fail for complicated or non-convex objects.

2.8 Hypothesis Verification

Historically, verification has meant several different things in the context of vision. The fundamental notion is that of confirming the existence of an oriented object, but this is often reduced to merely confirming the presence of a few object features.

Typical verification methods predict image features (e.g. lines) given the model and current state of analysis. These suggest a hypothesis, which is then strengthened or rejected according to the presence or absence of confirming evidence. (e.g. [FAL72]). Additionally, the discrepancy between the observed and predicted position can be used to refine the position estimates ([YIN84]).

Verification has mainly been used in the context of 2D industrial scenes, as in parts location systems (e.g. [LUX83]). Object silhouettes are most often used, because they make the object contours explicit; however, edge detected grey level images also produce similar information. The most common verification feature is the edge, and usually just the straight edge is used. Its main advantages are that its shape, location and orientation make its image location easy to predict and make it unlikely that scene coincidences could have created a similar structure. Prediction also allows more sensitive edge detection ([SHI75], [YAC79]), when searching for confirming evidence. Bolles ([BOL80]) used small slots and holes at given distances from a test feature in silhouette images as verification features.

In 2D scenes overlapping parts weakens the utility of contours, because only part of each object's outline is visible, which is also joined with those of the other objects in the pile. Since most 2D recognition systems are dependent on contours, this produces a serious loss of information. Yin ([YIN81]) hypothesized objects based on visible corners and linear features and verified them by ensuring that all unlocated corners were within the contours of the collected mass.

Verification in 3D scenes is a topic that has not received much attention. Some work similar to the line verification in 2D, but in the context of blocks world scenes, has been done by Falk ([FAL72]) and Shirai ([SHI75]). The presence or absence of searched-for lines confirmed or refuted the hypotheses. ACRONYM's ([BRO81]) prediction graph informs on the observable features, their appearance and their inter-relationship for more complicated objects (e.g. wide-bodied airplanes), but it was used for constraining object identity in a graph matching regime, rather than for direct search.

Hogg ([HOG84]) verified generalized cylinder model positions in 3D by counting oriented edge points within image boxes. The boxes were predicted using the projected outlines of generalized cylinders.

Occlusion is an even greater problem in three dimensions, as scenes have natural depth and hence objects will often self-obscure as well as obscure each other. Brooks ([BRO81]) suggested that a model-based geometrical reasoning vision system could predict what features will be self-obscured from a given viewpoint. Other work has been in the context of the blocks world scene analysis. Occlusion hypotheses are verified by detecting single tee junctions to signal the start of occlusion (e.g. [WAL75]) and pairs of tees indicate which edges should be associated (e.g. [GUZ67]).

Chapter 3

Surface Data as Input for Recognition

This chapter describes the visual data inputs used in the research. It also motivates the use of surfaces as the primary scene representation for object recognition. Surface data is initially unorganized depth and surface orientation, which is then segmented into surface regions suitable for recognition. The key contribution of the chapter is a proposal for segmenting the raw data into regions of approximately uniform surface curvature.

3.1 Why Use Surfaces for Recognition?

It was Marr, in advocating the $2\frac{1}{2}$D sketch as an intermediate representation ([MAR82]), who brought surfaces into focus. Vision is obviously a complicated process, and most computer-based systems have been incapable of coping with both the system and scene complexity. The importance of Marr's proposal lies in having a reconstructed surface representation as a significant intermediate entity in the image understanding process. This decision laid the foundation for a theory of vision, by splitting the rest of vision into those processes that contribute to the creation of the $2\frac{1}{2}$D sketch and those that use its information. A considerable proportion of vision research is currently involved in generating the $2\frac{1}{2}$D sketch (or equivalent surface representations). This thesis addresses the problem of what to do after the sketch is produced.

Five questions are discussed in the following subsections to motivate using surfaces:

1. What do we mean by surface information?

2. From what sources can we expect to gather this type of data?

3. What use can we make of a surface representation?

4. How can this information help overcome some current recognition problems?

5. Why not use other representations, such as boundaries, image regions, or solids?

Surface Images

A surface image is like Marr's $2\frac{1}{2}$D sketch ([MAR82]). This representation contains several types of information: surface depth and orientation, surface connectivity and various discontinuity segmentation labels (e.g. depth and surface orientation). A two dimensional "surface image" representation is used, with surface data pixels aligned with their corresponding intensity data pixels. Thus surface connectivity is implicit in the image array and each pixel has depth, orientation and any segmentation information.

The term "shape segmentation" is used loosely throughout this thesis, though section 3.2 proposes a set of criteria for segmentation. The intuitive notion desired is to segment the surface into regions of nearly uniform shape. Likely points of segmentation are at orientation, curvature and curvature rate of change discontinuities.

The precise placement of segmentation boundaries on surfaces (or even of their presence at all) varies significantly as a function of scale. What appears to be a smooth surface at one scale may show significant shape changes at finer scales (e.g. an orange skin with pores or a thumb with fingerprints). As a result,

both object models and image analysis should have a scale-based hierarchical description. This step was not taken in the research. Instead, only a single level of segmentation was used. This decision allows recognition to occur only when the object appears in a resolution range corresponding to that of the object model. Hence, the image and model segmentations will be required to have roughly corresponding surface regions, though not precisely corresponding boundaries.

Surface Data Sources

The research presented here is based on the assumption that there will soon be practical means for producing surface images. At present, there are four major approaches being researched.

- direct sensing – These include laser and sonar range finders, structured light methods ([POP75]) and triangulation methods ([KAN81a], [PIP82]). The point is to directly stimulate the environment, to take a more unambiguous measurement.

- stereometric methods – These are geometric stereo methods, based on matching of corresponding features in several images ([MAR82], [MAY80], [GRI81]).

- image property analysis – The most notable of these methods are the shape from shading techniques ([HOR75], [WOO79], [PEN82]). Other methods include shape from texture ([WIT80], [STE79], [PEN83], [OHT81]) or shape from surface contour ([BRA83], [STE83], [FIS83]).

- motion analysis – Optical flow analysis also provides the connectivity, orientation and discontinuity information ([HOR81], [NAG83], [RIE83], [CLO80], [PRA79]).

There are variations in the exact outputs of each of these techniques; but, most provide the required data. Some attributes must be derived from measured values (e.g. orientation by differentiating depth).

Use of Surface Representations

The surface image is a mixed viewer and object-centered representation, under general viewpoint assumptions ([COW83]). The depth and orientation values are object properties, but are expressed in a viewer-centered reference frame. Shape discontinuities are linked to object surface properties without viewpoint consideration. Depth discontinuities are dependent on both surface shape and viewpoint as is feature size and extremal boundary shape. In summary, most data is directly about the object surface, but is expressed relative to the viewer and which data features are observed is dependent on viewpoint.

A surface image makes useful information explicit. Surface orientation can be directly compared with that of a model surface, thus reducing the model-data registration problem by two degrees of freedom. (Instead of three potential rotations, only a single rotation about the aligned surface axis remains.) As surface regions are matched to model surfaces (assuming ideal models and data), the spatial relationship is unambiguous: in unoccluded situations, it is a six parameter geometric transformation plus a projection onto the image plane. For a single surface segment, the depth, surface orientation and image plane position data can be used to estimate the transformation parameters directly as a function of the undetermined rotation parameter (chapter 10).

Depth discontinuities explicitly denote the occlusion of more distant surfaces, hence simplifying the model matching and verification process.

The surface image provides surface segments that, assuming scale equivalences, correspond directly with model segments, especially since boundaries are only used for rough surface shape alignments. This is important because symbolic correspondences allow direct instantiation of discrete object models, which then facilitate geometrical inversions of the image formation process.

Marr and Nishihara ([MAR78]) argued for proceeding from the surface data in the $2\frac{1}{2}$D sketch to a generalized cylinder representation of the scene, by analysis of the axes of elongation and occluding contours of regions. This transformation ignored most of the information available in the $2\frac{1}{2}$D sketch , which is

too useful to be thrown away. Image axes are useful for elongated objects, but surface information must be a significant factor in the precise recognition of an object. The various axes about a head might invoke the general head model, but it is the surface shapes that really establish the person's identity.

Overcoming Some Current Recognition Problems

There are five recognition problems that surface images help overcome:

- interpreting two dimensional information using three dimensional objects,
- coping with occlusion,
- coping with noise,
- recognizing non-blocks world objects, and
- quickly invoking a reasonable model for an unidentified object from a large set of possibilities.

Viewed objects are often obscured by nearer objects and the ensuing loss of data causes recognition programs to fail. Programs based on detecting configurations of a few features are less sensitive to this problem (e.g. [BOL80]), but are also unintelligent recognition programs, and can thus be fooled by roughly similar objects or when the features are missing. (However, reductive methods are currently necessary in practical vision systems.) Such methods are useful to the formation of descriptions (chapter 8) needed for invoking the model (chapter 9), but are not properly complete recognition methods. Surface images provide two types of extra information that help overcome occlusion problems. First, occlusion boundaries are explicit, and thus denote where relevant information stops. Secondly, the presence of a closer surface provides evidence for why the information is missing, and hence where it may re-appear (i.e. on the "other side" of the obscuring surface).

Noise is omnipresent. Sensor imperfections, quantization errors, random fluctuations, surface shape texture and minor object imperfections are typical sources of data variation. A surface image segment is a more robust data element, because its size can average away some data variation (based on $O(n^2)$ data values as compared for $O(n)$ for linear features). The loss of information from using points or lines makes it difficult to determine the image to model correspondences correctly, whereas surfaces are larger and thus less ambiguous.

Connectivity (i.e. adjacency) of surfaces is largely guaranteed, and slight variations will not affect description. This contrasts with linear feature detection and description processes, in which noise causes loss of connectivity or wandering. If a linear feature based recognition process loses a edge, recognition may fail, or aspects of the object's shape will be lost. If a surface segment boundary is missing then there is just a larger surface, which would still be matchable (assuming the model has a full hierarchical scale-based surface description).

Selecting the correct model from the model database is difficult. Initially, what is needed is a description of the data suitable for triggering candidates from the database. As the input surface data is about the object's 3D surface, descriptions based on the data can be object-centered and so invoke models more effectively because of the shorter transformation distance between data and model.

Last, current recognition programs have problems with objects whose shapes are slightly more complicated than blocks. One cause of this has been a preoccupation with orientation discontinuity boundaries, which are noticeably lacking on many real objects and difficult to detect in intensity images. Using the actual surfaces as primitives extends the range of recognizable objects. Faugeras and Hebert ([FAU83]) used planar patch primitives (but not the patch boundaries) to successfully detect and orient an irregularly shaped object.

Other Input Data Representations

There are four major contenders for the primary input data representation: edges, image regions, surfaces and volumes.

Edges have been used extensively in previous vision systems. The key limitations of their use are:

- ambiguous scene interpretation (i.e. whether caused by occlusion, shadows, highlights, surface orientation discontinuities or reflectance changes),
- loss of data because of noise or low contrast, and
- image areas free from edges also contain information (e.g. shading).

Image regions are bounded segments of an intensity image. Their meaning, however, is ambiguous and their description is not sufficiently related to 3D objects. For example, Hanson and Riseman ([HAN80]) and Ohta ([OHT79]) segmented green image regions for tree boughs using color, yet there is no reason to assume that trees are the only green objects nor that contiguous green regions belong to the same object. Further, the segmentations lose all the detailed structure of the shape of the bough, which may be needed to identify the type of tree. They augmented the rough classification with general context relations, which assisted in the interpretation of the data. While this type of general information is important and useful for scene analysis, it is insufficiently precise and object-specific for identification (but is useful for model invocation). The conclusion is that image regions, without additional interpretation, are too weak and underconstrained.

The representation advocated here is that of the surface image region, which is a surface shape segmented portion of a surface image. It corresponds to a image region in an intensity image, but is segmented based on the 3D surface properties given in the surface image. The segmentation boundaries completely surround the region. The surface regions thus correspond directly with discrete portions of the object surface and have full 3D shape and placement information.

Volumetric primitives seem to be useful, as discussed by Marr ([MAR82]) and Brooks ([BRO81]) in their advocation of generalized cylinders. These solids are formed by sweeping a cross-section along an axis and represent elongated structures well. For volumes with shapes other than something like generalized cylinders, the descriptions are largely limited to CAD methods: explicit space-filling primitives or bounding surfaces.

The generalized cylinder and CAD space-filling primitive approaches have problems with deducing volume descriptions from visible surface data. What is perceived is a surface, yet what is represented is a volume, and there is no simple transformation from the surface to the solid under most representations. Marr ([MAR82]) showed that generalized cylinders were a logical primitive because these are the objects with planar contour generators from all points of view (along with a few other conditions) and so are natural interpretations for pairs of extended occluding boundaries. Unfortunately, few objects meet the conditions. CAD boundary description techniques depend on finding the boundaries, which leads back to the problems associated with edges. The explicit space filling approach is insufficiently compact, nor does it have the power to easily support many deductions.

3.2 The Labeled Segmented Surface Image

The labeled segmented surface image is the primary data input for the recognition process described in this thesis. Each of these terms is discussed in greater detail below but, by way of introduction, a surface image is a two-dimensional representation whose geometry arises from a projective transformation of the three-dimensional scene, and whose contents describe properties of the visible surfaces in the scene. Segmentation is considered here to be a partitioning of the the surface into regions of uniform properties (namely curvature and continuity) and labeling is an identification of the type of boundary between surface regions (e.g. whether occluding or shape).

Justifications for using surface images were discussed in the previous section and summarize to:

- the surface image gives direct information about the observed scene, and
- the variety of information makes the estimation of object properties easier.

Surface Image Data

A surface image is similar in structure to the traditional intensity image. The positional relationship between the scene and the image is described by projective geometry. What differs is the content of the surface image. Surface images record surface properties, rather than intensity properties, such as absolute depth and surface orientation in the global coordinate system. In this way, they are similar to intrinsic images ([BAR78]), except that here the information is solely related to the surface shape, and not to reflectance or externals. This eliminates surface markings, shadows, highlights, shading and other illumination and observer dependent effects from the information (which is also important, but is not considered in this research).

A second similar representation is Marr's $2\frac{1}{2}$D sketch ([MAR82]). This represents mainly surface orientation, relative depth and labeled boundaries. The difference with the work here is that Marr hypothesized that relative rather than absolute (viewer-centered) depth should be represented. Relative depth gives surface ordering and rough relative sizes for adjacent structures, whereas absolute distances gives these and also simplifies the calculation of absolute quantities, such as length, area and curvature.

Surface orientation allows calculation of the surface shape class (e.g. planar, singly-curved or doubly-curved), and correction of slant and curvature distortions of perceived surface area and elongation (chapter 8). The relative orientations between structures give strong constraints on the identity of their superobject and its orientation (chapter 10). Absolute distance measurements allow the calculation of absolute sizes.

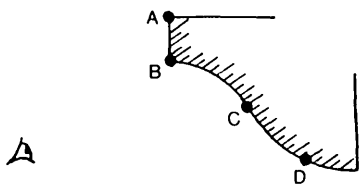


Figure 3–1: A 2D Surface Image – Viewer and Scene Geometry

There is redundancy in this information, in that surface orientation is, in principle, derivable from surface distance. However, this research is more concerned with how to use the information than how it was acquired or how to make it robust. The quality of the information is also important, but this aspect was not considered either – it was felt that the general principles were more important to investigate initially than robust practices.

Figure 3–1 shows the viewer-scene relationship of a two-dimensional "surface image". Figure 3–2 shows the absolute depth from the viewer for a 2D surface image. Figure 3–3 shows the surface orientation vectors associated with the corresponding image points.

Segmentation

The surface image is segmented into significant regions, resulting in a set of connected boundary segments that partition the whole surface image. What "significant" means has not been agreed on (e.g. [WIT83b], [LOW84]), and lit-

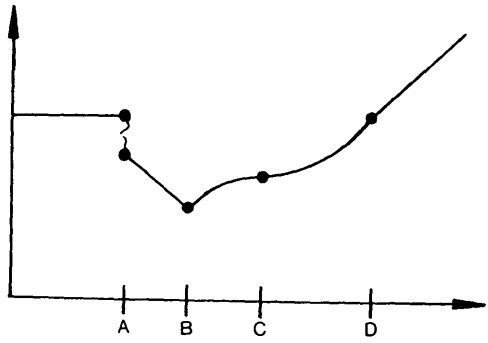


Figure 3–2: 2D Depth Component

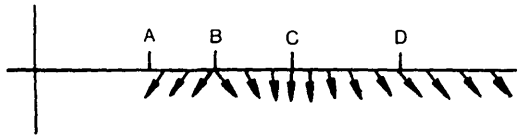


Figure 3–3: 2D Orientation Component (Vectors)

tle has been written on it in the context of surface representations. For the purposes of this research, it means **surface image regions corresponding to connected object surface regions with approximately uniform curvature and not otherwise terminated by a surface shape boundary.** The goal of this segmentation is to produce uniform regions whose shape can be directly compared to that of model surfaces. Some proposals for this were presented in ([FIS85a]) and are summarized below. In particular, the following are assumed to cause segmentation:

C1 occluding boundaries – points where a depth discontinuity occurs,

C2 surface orientation boundaries – points where a surface orientation discontinuity occurs,

C3 curvature magnitude boundaries – where a discontinuity in surface curvature exceeds a scale-related threshold, and

C4 curvature direction boundaries – where the direction of surface curvature has a discontinuity. This includes the change from concave to convex surfaces.

Other researchers have also considered these features for surface descriptions ([BRA84a]), rather than for segmentation.

The four segmentation rules listed above are minimum constraints. The first rule is obvious because, at a given scale, surface portions separated by depth should not be in the same segment. The second rule applies at folds in surfaces or where two surfaces join. Intuitively, the two sections are considered separate surfaces, so they should be segmented. The third and fourth rules are less intuitive and are illustrated in figures 3-4 and 3-7 below. The first example shows a cross section of a planar surface changing into a uniformly curved one. Neither of the first two rules applies, but one would clearly like a segmentation point near point X. However, it is not clear what to do in figure 3-5, where the curvature changes continuously. In figure 3-6, there are four points where the

Figure 3–4: Segmentation at Curvature Magnitude Change in 2D

curvature changes, but the exact location of the points is uncertain. Figure 3–7 shows a change in the curvature direction vector that causes a segmentation point as given by the fourth rule. Figures 3–2 and 3–3 show the various segmentation points for that example selected by this process (at points A,B,C and D). The corresponding segmentation types are:

point	type
A	depth
B	orientation
C	curvature direction
D	curvature magnitude

These results seem reasonable in two dimensions, but for surfaces it is not obvious how to generalize them. The step and orientation discontinuity rules generalize. However, the curvature rule needs extension because of the two dimensions of curvature. It was proposed ([FIS85a]) that discontinuity segmentation occurs along and across 3D curves lying on the surface (e.g. the principle directions).

Figure 3–5: No Segmentation on Continuous Changes

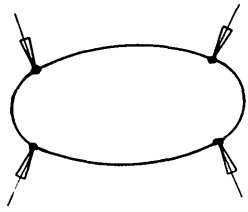


Figure 3–6: Segmentation at another Curvature Magnitude Change in 2D

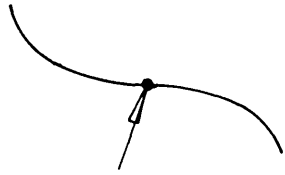


Figure 3–7: Segmentation at Curvature Direction Change in 2D

The rules segment surfaces into the six classes illustrated in figure 3–8. The class labels become symbolic descriptions for the surface.

The theoretical grounds for these conditions are not settled, but the following general principles seem reasonable. The segmentation should produce connected regions of similar character, having all curvature magnitudes roughly the same and in the same direction (i.e. segment the surface regions according to surface class). Further, the segmentations should be stable to viewpoint and minor variations in object shape, and should result in unique segmentations.

Figure 3–9 shows the segmentation of a sausage image. The segmentation produces four object surfaces (two hemispherical ends, a nearly cylindrical "back", and a saddle surface "belly") plus the background planar surface. The segmentation between the back and belly occurs because the surface changes from ellipsoidal to hyperboloidal. These are simple segments, stable to minor changes in the sausage's shape (assuming the same scale of analysis is maintained), and all surfaces are members of the six surface classes. Appendix figures

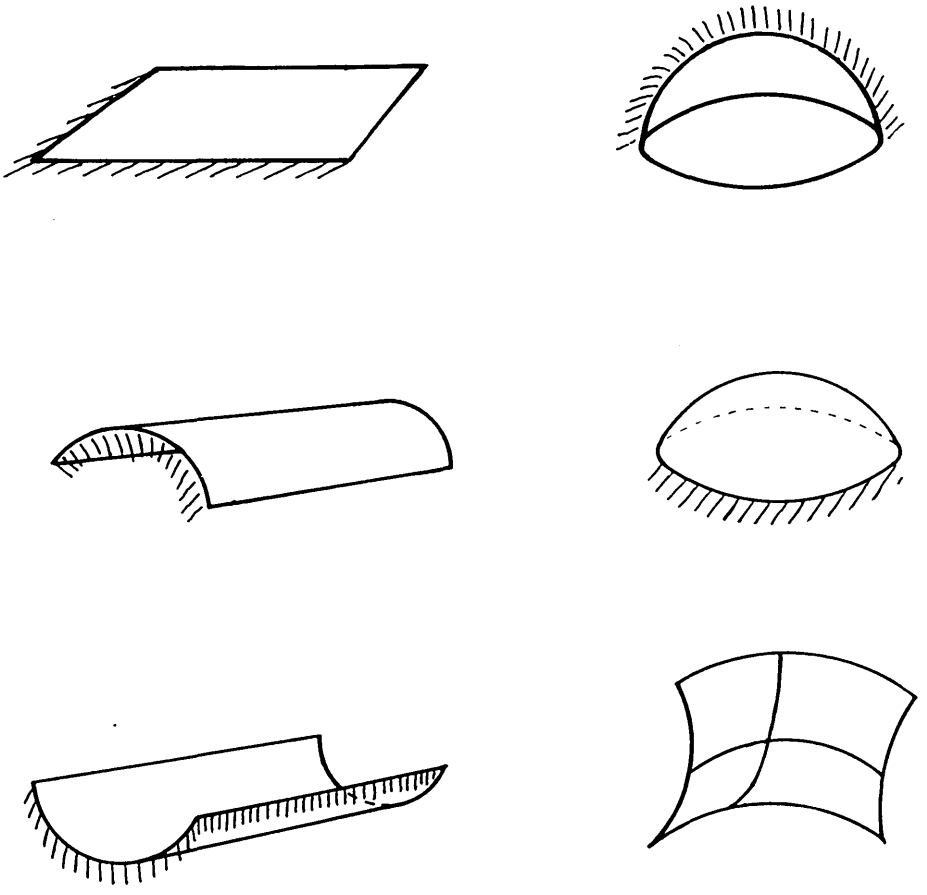


Figure 3–8: The Six Curvature Classes

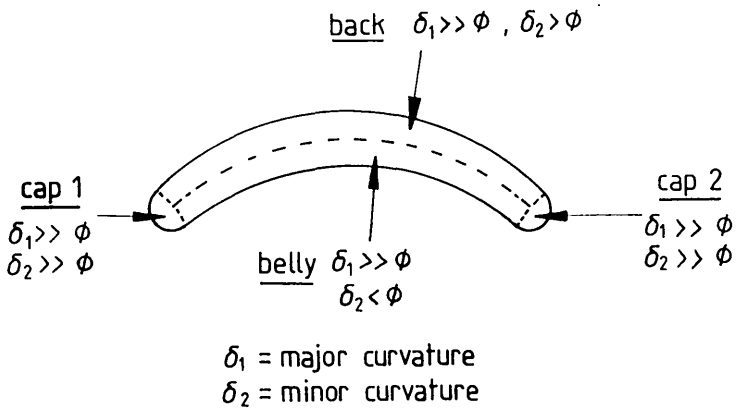


Figure 3–9: Segmentation of a Sausage

A–6 and A–15 show the surface patches produced by the segmentation criteria for the two test images.

Scale affects segmentation because some shape variations are insignificant when compared to the size of objects considered. In particular, less pronounced shape segmentations will disappear into insignificance as the scale of analysis grows. For example, a field of grain is flat when viewed from 1000 meters height, undulates gently from 100 meters, looks a bit ragged at 10 meters and separates into individual clumps and stalks at 1 meter. No "true" segmentation boundaries exist, so criteria for a reasonable segmentation are difficult to formulate. (Witkin ([WIT83a]) has suggested a stability criterion for scale-based segmentations of one dimensional signals.) The surfaces examined in this research were chosen to be uniformly segmentable to avoid this issue.

Another problem with segmentation at a single level of scale is it allows regions to not be fully surrounded by segmentation boundaries. This problem is believed to disappear when multiple scale analysis is used, by linking segmen-

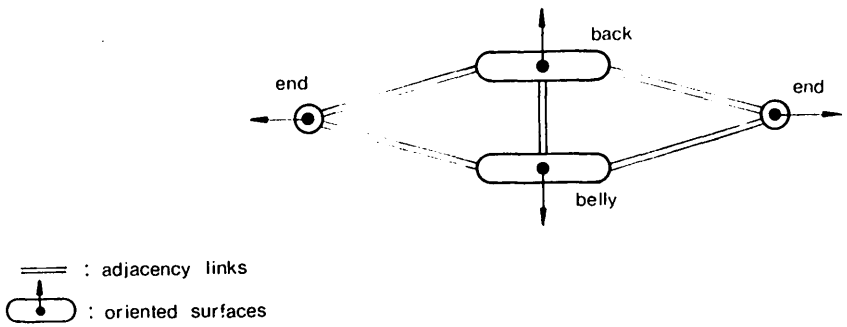


Figure 3–10: Representing the Segmented Sausage (From Figure 3-9)

tation boundaries from other scales. Practical problems undoubtedly exist as well.

Because the criteria are object-centered, they give unique segmentation, independent of viewpoint. Unique segmentations are preferred because they simplify model invocation (chapter 9) and hypothesis completion (chapter 10) by allowing a one-to-one model to segmented surface matching. Hence, segmentations that are insensitive to minor variations in object shape or segmentation scale are preferred. This does not imply that the segmentation boundaries must remain constant. For some ranges of scale, the sausage's boundaries (figure 3-9) will move slightly, but not introduce a new segmented surface. Invocation and matching avoid the boundary movement effects by emphasizing the spatial relationships between surfaces (e.g. adjacency and relative orientation) and not the position of intervening boundaries. The sausage example can thus be represented by the graph of figure 3–10. Here, the nodes represent the surfaces and are labeled by the surface class, curvature values and nominal orientations. The links denote adjacency.

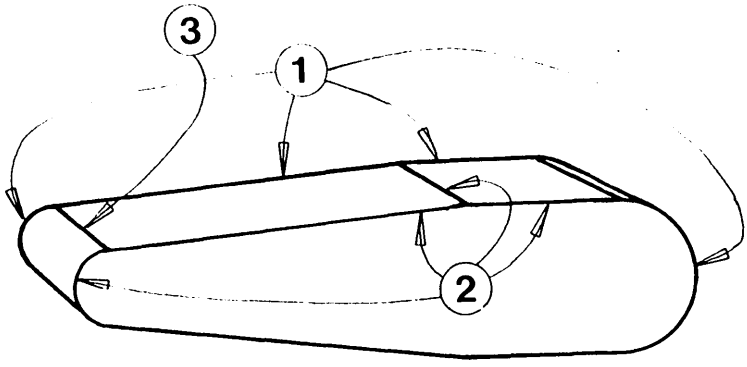


Figure 3–11: Example of Segmentation

Further, as the segmentation criteria is object-centered, the criteria can be applied to both model and data. Then, the model and data will have closely corresponding descriptions, which facilitates matching. Figure 3–11 shows the segmented robot upper arm model, with the type of segmentation boundaries noted.

In use, the exact type of shape segmentation boundary is not important, though distinctions between shape and obscuring boundaries are still important. Fisher ([FIS85a]) showed that shape boundary type may change as the analysis scale changes, so the particular label is not important, just its existence and location. However, the shape of the boundary helps identify particular surfaces, both during model invocation (chapter 9) and hypothesis completion (chapter 10).

Segmentation, as described here, assumes that the complete surface has been reconstructed. Blake (in discussion) suggested that some segmentation must be done before reconstruction, as knowledge of the occluding and shape boundaries is needed to control reconstruction ([TER83]). The low level data collection

processes (e.g. stereo) are likely to give more data near discontinuities, so segmentation can proceed at these points, and interior surface interpolation can follow this step. An alternative approach uses only the sparse surface data to segment the surface. This replaces actual surface reconstruction by a notional one.

Labeling

With the segmentation processes described in the previous subsection, the point and boundary labeling problem becomes trivial. The purpose of the labeling is to designate which boundaries result from the shape segmentations, and which result from occlusion. As discussed previously, the particular type of shape segmentation boundary is probably not important, as scale changes can change the labeling. However, the different types are recorded for completeness. Occlusion boundaries are further distinguished into the boundary lying on a closer obscuring surface and that lying on a distant surface. A further distinction is made for the curvature segmentation types. If one travels along a curve (e.g. a line of curvature) on a surface, then a discontinuity point is signaled if there is a significant change in curvature either along the curve or perpendicular to the curve. The two types are distinguished here, and a segmentation curve is made up of the points. This also applies to curvature direction. In summary, the full set of segmentation boundary labels is:

- front-side-obscuring
- back-side-obscuring
- surface-orientation
- surface-curvature-magnitude-along-transversal
- surface-curvature-magnitude-across-transversal
- surface-curvature-direction-along-transversal

- surface-curvature-direction-across-transversal

and the full set of segmentation point labels (for segmenting curves) is:

- boundary-orientation
- boundary-curvature-magnitude
- boundary-curvature-direction

One boundary labeling problem was encountered in test image 1 (see figure A–6). Where regions 8, 16 and 29 meet at the left of the robot shoulder small surface, the three boundary segments are all surface orientation discontinuity boundaries. To segment the boundaries around the small surface, a boundary-orientation discontinuity point was placed on the curve. This point then also segments the boundary crossing the top of the robot base. The conclusion is that boundary segmentation must be individual to each surface.

Inputs Used in the Research

The surface information used in this research is the distance to and surface orientation at the corresponding scene points. The distance is recorded from the viewer, but because:

- perspective projection was used,
- the objects were distant from the viewer, and
- the field of view small

the measured distance was also approximately the perpendicular distance from the viewer plane. The surface orientation was recorded as a unit (P,Q,R) vector for each measured point, in world coordinates. The orientation information should be in viewer-centered coordinates rather than in world coordinates, but because world coordinates are readily convertible to camera coordinates which

are then convertible to viewer coordinates (assuming the camera's position is known), the distinction is unimportant here. The use of world coordinates simplified the manual measurement process.

Because the geometry of the surface image is the same as that of an intensity image, an intensity image was used to prepare the initial input. From this image, all relevant surface regions and labeled boundaries were extracted, by hand, according to the criteria described previously. The geometry of the segmenting boundaries was maintained by using a registered intensity image as the template. Then, associated with pixels corresponding to key scene points, the distances to and surface orientations at those points were recorded. The labeled boundaries and measured points were the data inputs into the processes described in this thesis.

Because the orientation was collected in world coordinates, a single measurement point was sufficient for planar surfaces (figure 3-12). For curved surfaces, several points were used to estimate the curvature, but it turned out that not many were needed to give acceptable results. For most cylindrical surfaces six or nine well distributed points were enough (see figure 3-13). As the measurements were made by hand, the angular accuracy was about 0.1 radians and distance accuracy about 1 cm (estimated), but the errors proved to be unimportant.

Those processes that used the surface information directly (e.g. for computing surface curvature) assumed that the distance and orientation information was dense over the whole image. Dense data values were interpolated from values of nearby measured points. The interpolation used a $1/R^2$ image distance weighting that tended to flatten the interpolated surface in the region of the data points, but had the benefit of emphasizing data points closest to the test point. (A better process would have been a surface fitting approach (e.g. [TER83]). The interpolation used only those points from within the segmented surface region, which was appropriate because the regions were selected for having uniform curvature class. Appendix A shows the input data for each test scene in greater detail, recording scene depth, x, y, and z components of surface orientation,

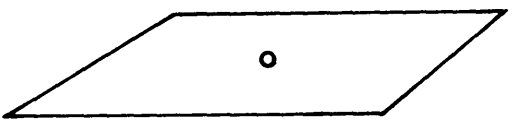


Figure 3–12: Location of Measurement Points for Plane

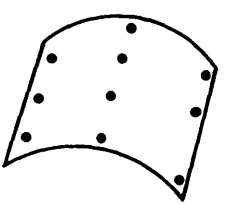


Figure 3–13: Location of Measurement Points for Cylinder

segmented surface image regions and the labels on the different segmentation boundaries.

The Region Graph

The region image information is organized into a graph structure. No information is lost in the transformations between representations, because of explicit linking back to the input data structures (even if there is some loss of information in the generalization at any particular level).

The labeled, segmented surface image has the following properties:

1. Regions are connected sets of surface points.

2. Boundary segments are connected sets of boundary points.

3. All points in one boundary segment have the same type.

4. Every region is totally bounded by a connected chain of boundary segments.

5. If one region is the front side of an obscuring boundary, the adjacent region is the back side.

These properties allow one to create a graph structure representing the input image with nodes for surface image regions and boundaries and links for adjacency. The properties of this graph are:

1. Region nodes represent complete image regions.

2. Boundary nodes represent complete boundary segments.

3. Chains of boundary nodes link connecting boundary segments.

4. Region nodes link to chains of boundary nodes that isolate them from other regions.

5. Region nodes corresponding to adjacent regions have adjacency links.

The computation that makes this transformation is a trivial boundary tracking and graph linking process. The only interesting point is that before tracking, the original segmented surface image may need to be preprocessed. The original image may have large gaps between identified surface regions. Before boundary tracking, these gaps have to be shrunk to single pixel boundaries, with corresponding region extensions. (These extensions have surface orientation information deleted to prevent conflicts when crossing the surface boundaries.) This action was not needed for the hand segmented test cases in this thesis, but real data is likely to require something like this.

3.3 Scene to Image Geometry

The information needed for understanding the three dimensional character of a scene comes from geometrical relationships between the scene and image as well as the depth and orientation information in the surface image. This topic is covered briefly in this section.

The contents of the surface image were discussed in the previous section, but the relationship between the scene and the image was not. This needs to be considered as three dimensional information is also extracted from the geometry of the image. Notionally, the scene is a set of visible surfaces with depth, orientation and boundary information available at every point. This information is then geometrically transformed to produce the image. The two aspects of the transformation are how the global data points relate to the observer, and how these points relate to the image.

If the transformation between the global coordinate system to the observer coordinate system (i.e. camera coordinates) is C, then a point $\vec{x}$ in global coordinates is at $C^{-1}\vec{x}$ in camera coordinates. The ACRONYM program ([BRO81]) could derive the transformation from other constraints (e.g. the airplane is on the ground), but they would have to be sufficient to fully constrain C. As the

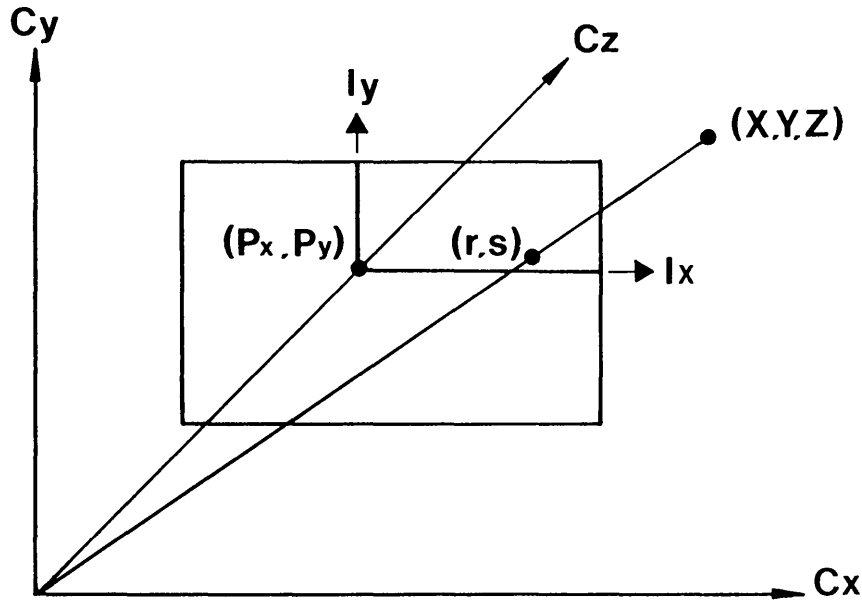


Figure 3–14: Camera Coordinates to Image Plane Geometry

global location is arbitrary, this merely relabels all scene locations, though perhaps more meaningfully.

The projection problem is summarized in figure 3–14. The optical axis is aligned with the $+C_z$ axis, and the image $+I_x$ and $+I_y$ axes are aligned with the $+C_x$ and $+C_y$ axes. Further, the optical axis passes through the point (P_x, P_y) in the image plane (usually the origin). Hence, the relationship between the camera coordinate point (x, y, z) and image plane point (r, s) is given by projective geometry:

$$r = P_x + \frac{g * x}{z}$$

$$s = P_y + \frac{g * y}{z}$$

where g is the scene distance to image distance conversion factor, as derived below (refer to figure 3-15).

Let:

w = width of the imaging surface, with N pixels

f = focal length of the lens system

d_1 = the distance from the lens to the point where the camera is focused

d_2 = the distance from the lens to the imaging surface

x = distance from the $C_y - C_z$ plane to the scene point

z = distance from the lens to the scene point

r' = distance on imaging surface in physical units (e.g. same units as d_1)

r = distance on the imaging surface in pixels

Then:

$$r' = d_2 * \frac{x}{z}$$

and

$$r = \frac{N}{w} * r' = \frac{N}{w} * d_2 * \frac{x}{z}$$

So, the distance conversion factor g is:

$$g = \frac{N}{w} * d_2$$

The lens equation gives:

$$\frac{1}{d_1} + \frac{1}{d_2} = \frac{1}{f}$$

Hence,

$$g = \frac{N}{w} * \frac{d_1 * f}{d_1 - f}$$

where N and w are constants that depend on the camera. For the experiments reported in this thesis,

$$\frac{N}{w} = 425$$

and

$$f = 1.9mm$$

To compensate for the 4:3 vertical expansion from a conventional TV camera, a compression of the input picture was necessary. With the camera used for the experiments, the scanning not perfectly linear so dimensions varied about 3% over the field of view.

This chapter justified surface data as being useful for 3D scene analysis. Its key contribution is a proposal for shape segmentation criteria.

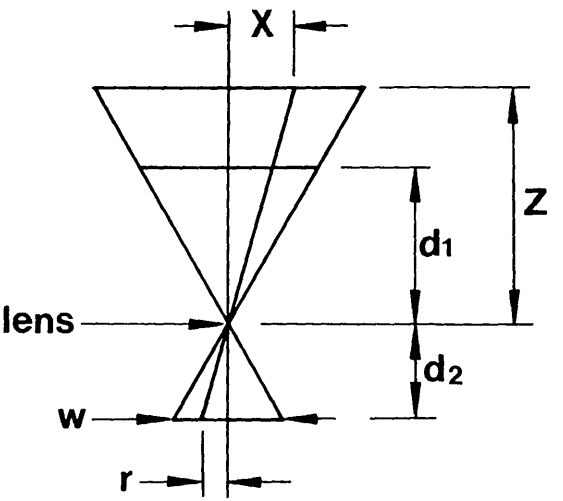


Figure 3–15: Focus Geometry

Chapter 4

A Model for Recognition Starting from Surface Information

Intuitively, object recognition is the isolation and identification of structure from the midst of other detail in an image of a scene. More formally, it is the assignment of a symbolic label to a group of features with the implication that those features could only belong to an object designated by that label. Hence, when we say we perceive (recognize) "John", we assert that there is a person, named "John", who accounts for all the perceived features, and that this person is likely to be at the specified location in the given scene.

Object recognition, when described like this, seems little different from a general concept-matching paradigm. So, what distinguishes it as a vision problem? The answer lies in the types of data, its acquisition and the representations of structures to be recognized. Vision is about perceiving structure ([WIT83b]) and this research addresses several visual aspects of how to do this:

- What are some of the relevant structures in the data?
- How is their appearance transformed by the visual process?
- How are they represented as a set of models?
- How are the models selected? and

- How is the data-model mapping established?

Object recognition also involves reasoning processes that map between the internal representations of the data, transforming the low levels of image description into the higher levels of object description. The transformations should reflect both the relationships between the representations and the constraints on the process. The most important constraints are those based on the physical properties of the visual domain and the consequent relationships between data elements.

The vision computation also has aspects in common with other cognitive processes – notably model invocation. Invocation selects candidate models to explain sets of data, a task which, in function, is no different from selecting "apple" as a unifying concept behind the phrase "devilish fruit". Invocation makes the inductive leap from data to explanation, but only in a suggestive sense, by computing from associations among symbolic descriptions.

Chapter 2 surveyed previous efforts at object recognition, and chapter 3 suggested how using surface information might improve recognition, both theoretically and practically. This chapter presents a computational structure for object recognition, based on those comments. Individual processes will be considered in later chapters, but the important relationships between them are discussed now. The first half of this chapter considers the problem of recognition in general, and the second half discusses the approach taken in this research. The four key topics covered are:

1. what is recognition?

2. when can an object be considered to have been identified?

3. what tasks does recognition require?

4. how can these be organized into a complete process?

4.1 The Nature of Recognition

Recognition (and perception) has received considerable philosophical investigation. Three key results are mentioned, as introduction to the recognition criteria proposed in this section.

(1) Perception is *interpretation* of raw sensory data. For example, I interpret a particular set of photons hitting my retina as the color green. As a result, perception is a internal phenomenon caused by external events. It transforms the sensory phenomena into a reference to a symbolic description. The perception may be directly related to the cause, but it may also be a misinterpretation, as with optical illusions.

(2) Interpretations are directly dependent on the theories about what is being perceived. Hence, a theory that treats all intensity discontinuities as instances of surface reflectance discontinuities will interpret shadows as unexplained or reflectance discontinuity phenomena.

(3) Identity is based on conceptual, rather than physical, relations. An office chair with all atoms replaced by equivalent atoms or that is damaged with a bent leg is still the same chair. Hence, any object with the appropriate properties could receive the corresponding identification.

So, philosophical theory implies that recognition has many weaknesses: the interpretations may be fallacious, not absolute and reductive. In practice, however, humans can effectively interpret unnatural or task-specific scenes (e.g. x-ray interpretation for tuberculosis detection) as well as natural and general ones (e.g. a tree against the sky). Hence, there must be many physical and conceptual constraints that restrict interpretation of both raw data as features, and the relation of these features to objects. This chapter investigates the role of the second category on visual interpretation.

How, then, is recognition understood here? Briefly, recognition is the production of symbolic descriptions. A description is an abstraction, as is stored

object knowledge. The production process is a series of transformations on sets of symbols producing other symbols. The transformations are guided (in practice) by physical, computational and efficiency constraints, as well as by observer history and by perceptual goals.

Transformations are implementation dependent, and may be erroneous, as when a simplified model of the ideal transformation is implemented. They can also make catastrophic errors when presented with unexpected inputs or when affected by distorting influences (e.g. optical, electrical or chemical). The notion of "transformation error" is not well founded, as the emphasis here is not on objective reality but on perceptual reality, and the perceptions now exist, "erroneous" or otherwise. The perceptions may be causally initiated by a physical world, but they may also be internally generated: mental imagery, dreams, illusions or "hallucination". These are all legitimate perceptions actable on by subsequent transformations; they are merely not "normal" interpretations.

Normal visual understanding is mediated by different description types over a sequence of transformations. The initial symbol may arrive by photon; later channels may be explicit (value, place or symbol encoded), implicit (connectionist) or external (e.g. distributed). The communication of symbols between processes (or back into the same process) is also subject to distorting transformations.

Object knowledge is limited to properties, these being symbolic assertions based on received descriptions. Knowledge is necessarily incomplete and, in practice, dependent on perceptual goals. There may be multiple overlapping and possibly inconsistent descriptions leading to the same object (symbol).

Identity is firstly linguistic: a chair is whatever is called a chair. Its second aspect is functional – an object has an identity only in relation to its place in the human world. Finally, identity implies properties, whether physical or mental.

An identification is a symbol whose associated properties are similar to those of the data, and is the output of a transformation, as discussed above. The properties (also symbols) compared may come from several different processes

at different stages of transformation. Similarity is not a well defined notion, and seems to relate to a conceptual distance relationship in the space of all described objects, but this is confused because the similarity evaluation is affected by perceptual goals. It is more likely to be based on functional than physical criteria.

This is the abstract view of recognition. The practical matters are now discussed: what is recognized and how.

Object Isolation

Traditionally, besides identification, recognition involves structure isolation, because naming requires objects to be named. This includes denoting what constitutes the object, where it is and what properties it has. Unfortunately, the isolation process depends on what is to be identified, in that what is relevant can be object-specific. However, this problem is mitigated because the number of general visual properties seems to be limited and there is hope of developing "first pass" grouping techniques that could be largely autonomous and model independent. So, part of a sound theory of recognition depends on developing methods for isolating specific classes of objects.

For example, a row of colinear dots could be declared a line, or a flat set of data a surface. A particular semi-distinct region of space in the middle of a face could be called a nose. The processes may not always be model independent, as the constellation Orion can be found and declared as distinguished in an otherwise random and overlapping star field.

This thesis considers isolation for three classes of structures: surfaces, rigidly connected solids and flexibly connected solids. The isolation process for surfaces (chapter 6) is almost immediate because surface regions are primitives of the segmented surface image. Complete surface regions are produced by an extension process. The isolation of solids is based on the adjacency of the component surfaces and the types of boundaries between them (chapter 7).

The Property Basis for Recognition

Chapter 2 concluded that correct properties were the traditional basis for recognition, the differences between approaches lying in the types of evidence used, the modeling of objects, what constituted adequate recognition and the algorithms for performing the recognition.

Some properties simply declare existence: a given feature is present. Other properties declare that a feature has a given attribute, or that groups of features have given relationships. Some properties are expressible by constraint ranges, such as the length of a line lies in a given range, or the relative orientation of two bodies is less than $\pi/2$. Object definition then becomes a listing of what properties an object should have, and what their valid ranges are.

In this research, surface and structure properties are the key types of evidence, and were chosen to characterize a large class of everyday objects. As 3D input data is used, a full 3D description of the object can be constructed and directly compared with the object model. The difficulty then arises in the construction of the 3D description. Fortunately, various constraints exist to help solve this problem (discussed below).

What distinguishes recognition in the sense used in this thesis is that it labels the data, and hence is able to reconstruct the image. While the description may be compressed (e.g. a "head"), there will be an associated prototypical model (organizing the properties) that could be used to recreate the image to the level of the description. This requires structural descriptions and features, and requires that identification be based on these properties, rather than on summary quantities such as volume or mass distribution.

Additionally, each object may have associated with it a variety of properties, some of which may be general (e.g. location) and some of which may be object specific (e.g. the size of a particular scratch on my hand). These properties may be necessary for the specific identifications (see next section), but at this point they are just considered to be secondary attributes. (The generic identity is primary.) Object size now becomes an associated attribute, whereas it could have

been treated as part of the identification itself (e.g. "10 inch pencil" becomes "pencil" with "length(10)").

Definition of Recognition

Recognition can be summarized as:

> recognition produces a fully instantiated, spatially located, described object hypothesis with direct correspondences to an isolated set of image data.

That is:

1. recognition isolates an object's features from other features,
2. recognition assigns an identity to the set of features, and
3. recognition attaches additional properties to the aggregate, such as spatial position, size, neighbor relations, etc.

Isolation is contingent on the class of the object and may depend on a partial identification for guidance. In chapter 7, some rules for autonomously isolating solids in a surface image are discussed, but other object phenomena require alternative visual isolation techniques. In the next section, the meaning of the identity predicate is defined. Finally, the choice of additional properties and how they should be deduced are left as open problems (except for object position).

"Fully instantiated" means that all object features predicted by the model have been accounted for, either with directly corresponding image data or with explanations for their absence. Because surfaces are the chosen model primitive, the key data are surface patches and the boundaries between them. The acceptable explanations for missing evidence are: the feature is on the back side of the object, the object obscures itself, or an unrelated object partially obscures the object.

This research investigates recognizing "human scale" rigidly or flexibly connected solids with uniform, large surfaces including: classroom chairs, most of a PUMA robot and a trash can. The types of scenes that these objects appear in are normal indoor somewhat cluttered work areas, with objects at various depths obscuring portions of other objects. Appendix B shows the objects recognized and appendix A shows the scenes analyzed.

Given these objects and scenes, four groups of physical constraints are needed:

- limitations on the surfaces and how they can be segmented and characterized,
- properties of solid objects; in particular, how the surfaces relate to the objects bounded by them,
- properties of scenes, including spatial occupancy and placement of objects, and
- properties of image formation and how the surfaces, objects and scenes affect the perceived view.

These are introduced in the appropriate chapters.

4.2 Criteria for Identification

The previous section postponed discussion of identification criteria. The proposed criterion is that the object has all the right properties and none of the wrong properties.

Perceptual goals influence the choice of properties used in identification. Unused information may allow distinct objects to acquire the same identity. If only the generic chair were modeled, then all chairs would be classified as the generic chair. This thesis has chosen to use only surface data, so distinctions based on color cannot be made.

The space of all possible objects may be sufficiently disjoint, by property, that the detection of only some properties may uniquely characterize. Efficient recognition may be possible by a parsimonious selection of these properties, but redundancy adds the certainty needed to cope with missing or erroneous data, much as the extra data bits in an error correcting code help partition the code space.

Conversely, a set of data might implicate several objects related through a relevant common generalization, such as similar yellow cars. Or, there may be no relevant generalization between alternative interpretations: (as the children's joke goes) Q:"What's grey, has four legs and a trunk?" A:"A mouse going on a holiday!".

Though the basic data may admit several interpretations, further associated properties may provide finer identifications, much as ACRONYM ([BRO81]) used additional constraints for class specialization.

While not all properties will be needed for a particular identification, some will be essential and recognition should require these when identifying an object. If some properties were optional, then the representation should be split into generic specializations. For example, a beer can would need to be cylindrical – its surfaces must enclose a volume. One could consider a picture of a beer can as if it were the original, but this is just a matter of choosing what properties are relevant. An object without some of these features, such as the printing from the beer can on a sheet of paper, may be reminiscent or suggestive of the object (chapter 9), but would not be acceptable as a proper instance.

There may also be properties that the object should not have, though this is a more obscure case. In part, these properties may contradict the object's function. Some care has to be applied here, because there are many properties that an object does not have and they should not have to be made explicit.

Most direct properties, like "the length cannot be less than 15 cm" can be rephrased as "the length must be at least 15 cm". Properties without natural complements seem to be rare, but exist – "subcomponent of" is one such prop-

erty. One might discriminate between two types of objects by stating that one has a particular subcomponent, and that the other does not and is otherwise identical. Failure to include the "not subcomponent of" condition would reduce the negative case to a generalization of the positive case, rather than an alternative. Examples of this are: (particular) a nail polish dot that distinguishes her from his toothbrush, (generic) a seat-back as the discriminator between a chair and a stool, and (temporal) the presence of a scar on a friend's face to distinguish before and after pictures.

Recognition takes place in a context – each perceptual system will have its own set of properties suitable for discriminating among its range of objects. In the toothbrush example, the absence of the mark distinguished one toothbrush in the home, but would not have been appropriate when still at the factory (among the other identical, unmarked, toothbrushes).

Finally, the number and sensitivity of the properties affects the degree to which objects are distinguished . The area-perimeter ratio distinguishes some 2D objects in a 2D vision context, even though it is an impoverished representation.

One problem with 3D scenes is missing data. In particular, objects can be partially obscured. But, because of redundant features, context and limited environments, identification is still often possible. On the other hand, there are also objects that cannot be distinguished without a more complete examination – such as an opened versus unopened beer can. If complete identification requires all properties to exist, the missing ones will need to be predicted, based on models. It is assumed here that all objects, generic or specific, have geometrical models that allow predictive analysis. Then, if the prediction process is reasonable and understands physical explanations for missing data (e.g. occlusion, known defects), the object will be consistent with the observed data, and hence have an acceptable identification.

The above discussion introduces most of the issues behind recognition, and is summarized here:

- the goals of recognition influence the distinguishable objects of the domain,

- the characterization of the domain may be rich enough to provide unique identifications even when some data is missing or erroneous,

- all appropriate properties should be necessary, with some observed and the rest deduced,

- some properties may be prohibited,

- multiple identifications may occur for the same object and additional properties may specialize them, and

- alternative properties may be used according to the recognition goals and sensory modalities.

The Basis for Identification Used in this Research

The preceding discussion concentrated on recognition in general. This section concludes with the details of the particular process implemented in this research.

There are three classes of recognized objects: surfaces, rigidly connected solids and flexibly connected rigid solids, and within these classes particular submembers are defined for recognition. The features used here are surfaces and 3D surface clusters.

All recognition systems must somehow be based on properties. What distinguishes this system is its choice of properties, which are structural and based on the 3D character of the features. For surfaces, the properties are:

1. surface shape,

2. boundary shape, and

3. boundary connectivity.

For solids, the properties are:

1. existence of identified subcomponents (surfaces or recursively defined subcomponents),

2. subcomponent 3D placement consistent with the model, and

3. subcomponent connectivity.

Boundary placement consistent with surface position (e.g. [FIS83]) is another (unimplemented) identification constraint. These properties give a powerful characterization of an object, in that they allow reconstruction of an object from its description.

The implemented specification and evaluation of these properties is not so precise and is somewhat distributed. For surfaces, model invocation (chapter 9) uses data shape properties to suggest models and candidates with insufficient correspondences are never invoked. Then, identity verification required only surface shape and absolute size to ensure correct surface identities. Extended model bases would necessitate more rigourous boundary shape comparisons.

For solids, the key features are surfaces and recursively defined sub-solids and the key properties are based on geometrical and topological relationships. These are the only properties considered here, but a complete vision system would include others, such as color, texture, more qualitative sizes and relationships and probably some functional relationships (e.g. the chair seat is "usable for sitting").

For the robot lower arm model, for example, the key features include the surfaces {left facing large side panel, right facing large side panel, ...} and the hand subobject. The geometrical relationships place the features relative to the nominal reference frame of the whole solid. The primary topological relationship is surface adjacency.

Because occlusion is certain, some properties will be missing. These can be deduced using located geometrical object models with image occlusion data (closer surfaces), as discussed in chapter 10. No prohibited features were needed here.

As with surfaces, solid property evaluation was distributed. Model invocation used subfeature identities and some solid properties (chapter 8). Hypothesis completion (chapter 10) required:

- finding a reference frame such that all data normals, surface curvature axes and subcomponent reference frames were approximately as predicted,
- features conformed to their predicted visibility, and
- all subfeatures were found or accounted for.

Verification (chapter 11) required the additional properties:

- no duplicate use of image features,
- all adjacent model features have image adjacency (or excuses), and
- features predicted to be partially obscured by features on another related assembly were observed as such.

In summary, an object can be considered recognized when it has all necessary and no prohibited properties and when it consistently "explains" a segmented set of image data. The structural properties chosen above seem to characterize many man-made objects and worked well for recognizing the objects in the test scenes.

4.3 Recognition Tasks

In the previous sections, recognition was defined, but how this was achieved was left unspecified. This section makes explicit the major sub-functions of the recognition process.

Recognition is based on comparing observed and deduced properties with those of a prototypical model. This definition immediately introduces five subtasks for the complete recognition process:

- finding the structures that have the properties,
- acquiring the properties,
- selecting a model for comparison,
- deducing any missing properties for the given model, and
- comparing the data with the model properties.

The deduction process needs the location and orientation of the object to predict obscured features and their properties, so this adds:

- estimating the object's location and orientation.

This, in turn, is based on inverting the geometrical relationships between data and model structures, which adds:

- making data (surface and subobject) to model correspondences.

The remainder of this section elaborates on the processes and the data flow dependencies that constrain their mutual relationship in the complete recognition computation context. Figure 4-1 shows the process sequence determined by the constraints. The numbers in parentheses give the corresponding chapter in the thesis.

Data segmentation and organization is most difficult and important. Its primary justification is that segmentation highlights the relevant features for the rest of the recognition process and produces, in a sense, a figure/ground separation. (Here, segmentation does not mean literal isolation of image regions, but rather perceptual grouping of related descriptions.) Properties between unrelated structure should not be computed, such as the angle between surface patches on separate objects. Otherwise, coincidences will invoke and possibly substantiate non-existent objects. Further, computational resources limit what properties can be feasibly computed. The number of computations is roughly

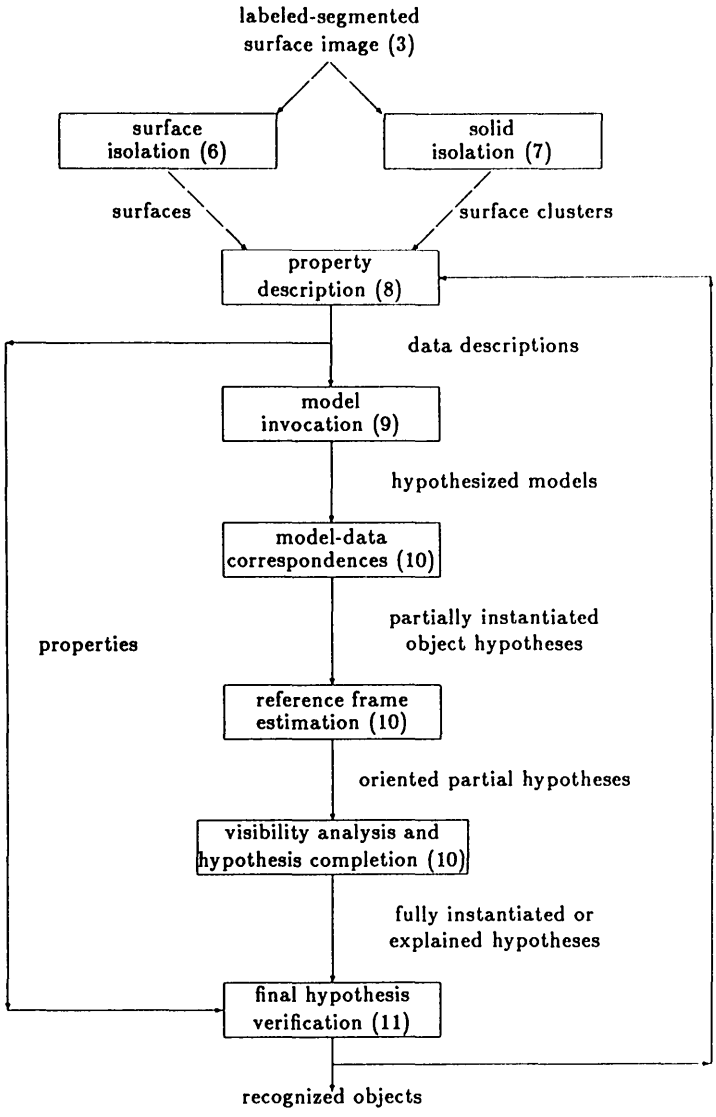


Figure 4-1: Sequence of Recognition Process Subtasks

proportional to the square of the number of features in the region of interest (assuming many properties are binary), so limiting that area is essential.

Organization is difficult, and the methods depend on the type of object being isolated. Boundaries and surface regions are presegmented in the input, and chapter 3 discussed how this might be achieved. Segmented groups of surfaces form surface clusters according to constraints discussed in chapter 7. The input to the process is the surface image and the outputs are a set of object surfaces and surface clusters.

Property extraction (chapter 8) creates the descriptions needed for model invocation and matching. This task uses the surfaces and surface clusters produced by the segmentation processes. The surface and boundary shapes are the key properties for surface regions. Feature sizes, spatial relationships and adjacency are the properties needed for solid recognition.

Model invocation (chapter 9) is essential because of the impossibility of selecting the correct model by sequential direct comparison with all known objects. These models have to be selected through suggestion because: (a) individual models may not exist (because of object variation or generic description) and (b) object flaws or variation, sensor noise and data loss lead to inexact model-data matchings. This process uses the segments and associated descriptions, and returns possible identities associated with the segments.

Model invocation is the purest embodiment of recognition. It is also the hallucination process – its outputs depend on its its inputs, but need not be verified or verifiable for the visual system to report results. Because we are interested in literal object recognition here, what follows after invocation is merely verification of the proposed hypothesis: the finding of evidence and ensuring of consistency.

Model invocation also makes correspondences between the model and data features by selection of high plausibility subcomponent evidence. This task takes sets of data features and a proposed model and returns hypotheses with feasible pairings. Surface correspondences are immediate because there are only two data elements of different types (surface and boundary). The solid correspon-

dences are also trivial because the matched substructures (surfaces or previously recognized solids) are also typed and are generally unique within the particular model.

The estimation of the solid and surface reference frames (chapter 10) is one goal of scene analysis and is also needed for making detailed metrical predictions during feature detection and occlusion analysis (below). This task has the advantage of an absolute constraint – given the estimated transformation, the predicted view of the model must correspond closely with the observed data. Starting from a partially instantiated hypothesis with paired data surface and nominal model surface orientations, it approximately inverts the data relationships to estimate the object orientation. Translations are estimated from the image displacements and depth information.

Hypothesis completion (chapter 10) finds evidence lost because of observer viewpoint (e.g. self-occlusion or closer, unrelated obscuring objects) or initial model-data pairing failures. The process starts from oriented and partially instantiated object hypotheses and produces fully instantiated hypotheses. Predictions are made about where features should be and their visibility, using a geometrical model of the object and estimates of its location and orientation. Self-obscured features can be excused as being not visible. The image can then be examined for direct evidence of other visible surfaces (i.e. finding a valid data surface, or other obscuring surfaces). The properties related to missing features are calculated implicitly – in the sense that the processes show that the data is consistent with the assumption of the obscured object.

Identity verification (chapter 11) is the crux of the recognition process and is distinct from hypothesis completion, which gathers all evidence for a hypothesis. Verification ensures that the hypothesis is correctly formed and has all of the required object properties. Hypothesis completion and verification follow the generate and test paradigm.

In this thesis, the two processes have been separated, but implementations of the processes need not be strictly sequential. Some verifications can be made when the necessary data becomes available, rather than when all evidence is

gathered. Surface identity verification follows assignment of the spatial reference frame transformation and involves rough shape comparisons. Solid identity verification occurs in several stages. The matching of features (surfaces and subobjects) is implicit in the assignment of data to the model. The matching of the geometrical information is implicit in the estimation of the object's reference frame because inconsistent data causes estimation failure. Finally, structure adjacency, structure duplication and visibility is then verified explicitly using the model.

The last two processes ensures that all required properties are held by the object through both model and knowledge directed processes. The object representation states what features are required and what inter-relationships they should have, and the program then goes through a variety of steps to confirm the properties. For example, given an initial estimate of the object's reference frame, some subsequent processes:

- use the model, object orientation and image formation knowledge to deduce invisible back-facing features,
- use the same to deduce fully or partially self-obscured features, and
- use the same to deduce, locate and match missing visible features.

This completes the characterization of recognition as an organized sequence of subtasks. Details of each subtask are discussed in chapters 6-11. Chapter 12 presents results for the recognition process as a whole, as applied to the test images (appendix A), with a critical discussion of its conception and performance.

Recognition Graph Summary

The entire recognition process creates a number of data structures, linked into a graph whose relationships are summarized in figure 4–2. This figure should be referred to while reading the remainder of this thesis.

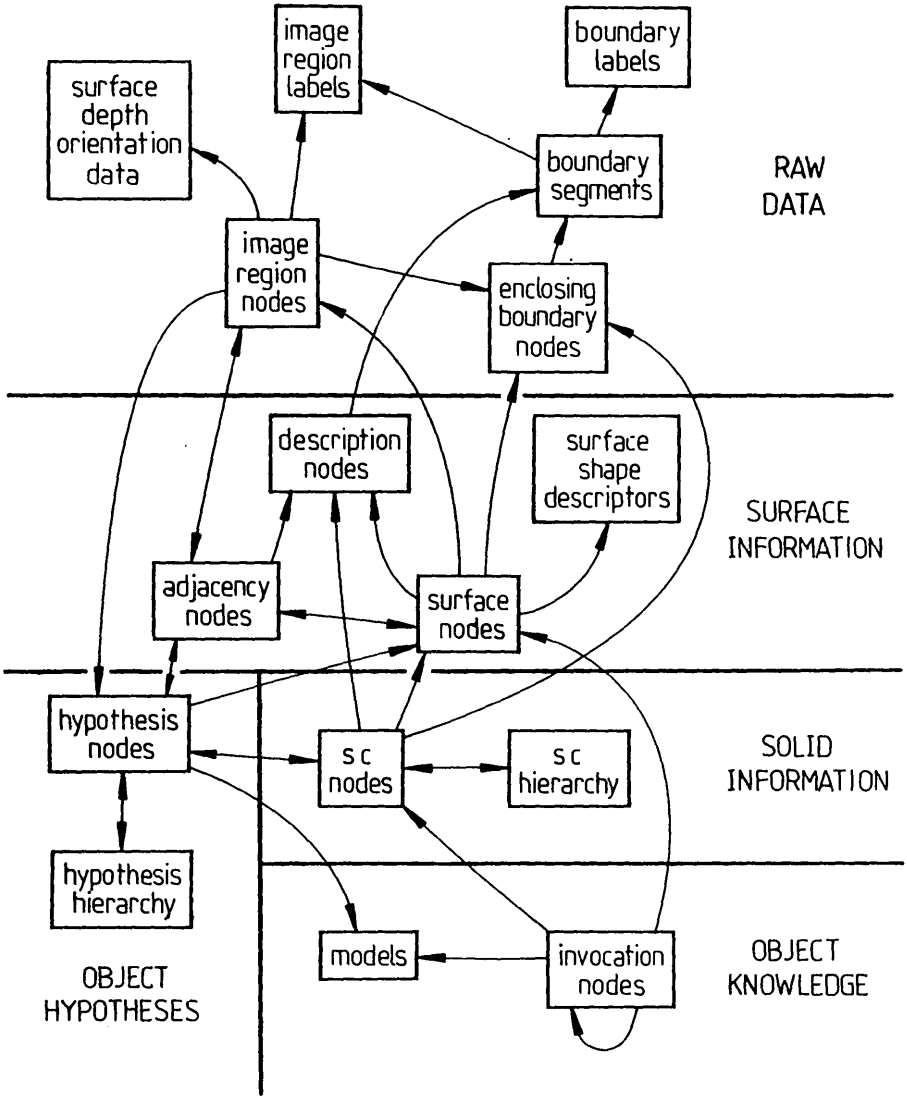


Figure 4–2: Summary of Data Structure Relationships

At the top of the diagram, the three bubbles "surface depth and orientation data", "image region labels" and "boundaries and labels" are the image data input into the process (chapter 3). The boundary points are linked into "boundary segments" which have the same label along their entire length. "Image region nodes" represent the individual surface image regions, and associate them with their "enclosing boundary". These are circularly linked boundary segments that completely enclose a structure. "Adjacency nodes" link adjacent region nodes and also link to "description nodes" that record (in this instance) which boundary separates the regions. The construction of these initial data structures is largely a reorganization of information in the raw data.

The image region nodes form the raw input into the "surface node" hypothesizing process (chapter 6). The surface nodes are also linked by "adjacency nodes" and an "enclosing boundary". In the description phase (chapter 8), properties of the surfaces are calculated and these are also recorded in description nodes. Also, the surface shape is estimated and recorded in the "surface shape descriptors".

The surface cluster formation process (chapter 7) aggregates the surfaces into groups recorded in the "surface cluster nodes". These nodes are organized into a "surface cluster inclusion hierarchy" linking larger enclosing or smaller enclosed surface clusters. The surface clusters also have properties recorded in description nodes and have an enclosing boundary.

Invocation (chapter 9) occurs in a plausibility network of "invocation nodes" linked by the structural relations given by the "models". Nodes exist linking model identities to structures (surface or surface cluster nodes). The invocation nodes link to each other to exchange plausibility among hypotheses.

When a model is invoked, "a hypothesis node" is created linking the model to its supporting evidence (surface and surface cluster nodes). Hypotheses representing objects are arranged in a generic and component hierarchy analogous to that of the models. Image region nodes link to the hypotheses that best explain them.

This chapter has:

- defined object recognition,
- explicated its identification criteria and
- summarized its computational structure.

The key contribution of the chapter is its specifying a thorough surface based image understanding process.

Chapter 5

Object Representation

To recognize objects, one must have an internal representation of these objects suitable for matching to descriptions. There are three processes that need object representations: model invocation (chapter 9), hypothesis construction (chapter 10), and hypothesis verification (chapter 11). This chapter reports on a surface based object representation method distinguished by its use of shape segmentable surfaces organized in a subcomponent hierarchy. It is argued that this representation provides the information needed for surface-based object recognition.

5.1 Requirements on Geometrical Body Models

The single most important representation is the geometrical body model, which is a structural description of the object. In a sense, the model summarizes what the the system knows about the appearance of the object. Paraphrasing Binford ([BIN82]): a capable vision system should know about objects, and how objects appear in images, rather than what types of images an object is likely to produce. From a geometrical body model, one can deduce what features will be seen from any particular viewpoint, and can determine under what circumstances a particular image relationship is consistent with the model. While a practical vision system may incorporate typical viewer-centered descriptions, these can be derived from the object-centered representation. Moreover, at times, unexpected

views of objects will be encountered, which will require the object-centered representation to predict object appearance.

The body model used here introduces a uniform level of description suitable for a large class of objects (especially man-made). Rather than have the model implementer decide what are the relevant features needed for recognition, the system can decide from the model itself, assuming the descriptive adequacy of the modeling system.

Modeling should emphasize the relevant aspects of objects. This thesis is concerned with model shape and structure, but not reflectance, so these are: surface shape and surface boundaries, inter-surface relationships (e.g. adjacency and relative orientation), surface-object relationships and subobject-object relationships. This information should be explicit or easily derivable, to simplify recognition. Further, because objects can take arbitrary spatial locations and orientations, the models need to be easily transformable.

Marr ([MAR82]) proposed five criteria for object representation:

1. *accessibility* – needed information in a model should be directly available, rather than derivable through heavy computation,

2. *scope* – a wide range of objects should be representable,

3. *uniqueness* – an object should have a unique representation,

4. *stability* – small variations in an object should not cause large variations in the model, and

5. *sensitivity* – detailed features should be represented as needed.

The principles are generally held here, except that the uniqueness criterion is weakened to become: an object should have only a few representations and these should be easily derivable from each other. An example where this might be required is a teapot body and spout grouped at one level of description with

the handle added at a larger level versus representing all features at the same level.

Based on arguments raised in chapter 2, the geometric models for the nearby object recognition system given in this thesis should:

- make surface information explicit,
- have three dimensional, transformable object-centered representations,
- represent solid and laminar objects,
- have geometrical subcomponent-object relationships, and
- have flexible attachments.

In addition, other specific requirements for geometrical information are discussed below.

Model invocation (chapter 9) is based on both direct evidence and associations. The direct evidence comes from comparing image data with model data. Additional data needed for invocation is:

- the size, shape and curvature parameters of individual surfaces and boundary segments,
- the adjacency of surfaces and their relative orientation,
- which boundary elements belong to each surface,
- which surfaces belong to each object,
- which substructures belong to each object, and
- what are typical visible configurations of components.

Model matching (chapter 10) instantiates invoked models by pairing image data or previously recognized objects with model components. It requires:

- the type of substructures needed and
- the geometrical relationships between the substructures and the object.

Object verification (chapter 11) ascertains the soundness of the instantiated models by ensuring the correct types of substructures were selected, that they have the correct geometrical relationship with each other and the whole, and that the assembled object forms a valid and compact solid. This requires:

- substructure types,
- substructure adjacency,
- surface shape class, and
- additional parameter constraints.

5.2 The Geometric Body Model

The geometric body model specifies the key primitive elements of an object representation, and shows how these elements are positioned relative to the object as a whole. As discussed chapter 2, there are a variety of potential primitives, but this research uses the surface patch for recognition purposes. Using the same primitives for both the models and data reduces the conceptual distance between the two, and thus simplifies matching. The segmented surface image (chapter 3) represents the currently analyzed scene using these surface patches. Hence, surface patch models should be a key object representation, and this thesis explores their use.

The primitive element of the model is the SURFACE, a one-sided bounded 2D (but not necessarily flat) structure defined in a 3D local reference frame. It has two primary characteristics – shape, and extent. The shape is defined by its surface class, the curvature axes and the curvature values. If any curvature is

given, the major curvature axis is defined by its endpoints (in 3D). The minor curvature axis (if any) is orthogonal to both the surface normal and the major curvature axis.

The surface classes are planar (no curvature), cylindrical (one direction of curvature) and ellipsoidal/hyperboloidal (two directions of curvature). The curvature values can be positive or negative, representing convex or concave principal curvatures about the curvature axes. While many other surface shape representations are used, this one was chosen because:

1. surface shape is characterized by two parameters only (the principal curvatures), and it is felt that these can be successfully estimated from image data (e.g. [BRA84a]). Further, it is felt that it will be difficult to estimate much more, and

2. even if the precise curvature values are not extractable, shape class should be, using a variety of shape cues (e.g. specularity, shadows, shading, etc.).

The surface description for a planar surface is:

PLANE

For a cylinder, it is:

CYLINDER[axis_end_1, axis_end_2, radius_1, radius_2]

where the two endpoints lie on the axis and the two radii are the cylinder radii at the respective endpoints (with intermediate values linearly interpolated). If the surface is convex, then the radii are positive, and the axis lies behind the surface. If the surface is concave, then the radii are negative and the axis lies in front of the surface. Figure 5–1 illustrates the surface definition.

Cylindrical surfaces that cover more than π rotation angle are problematic using the definition methods of this project. (A more sophisticated modeler could overcome the problems.) If the surface is a full 2π with no detail, then the surface is split into two halves; as only one half can be seen at a time, all

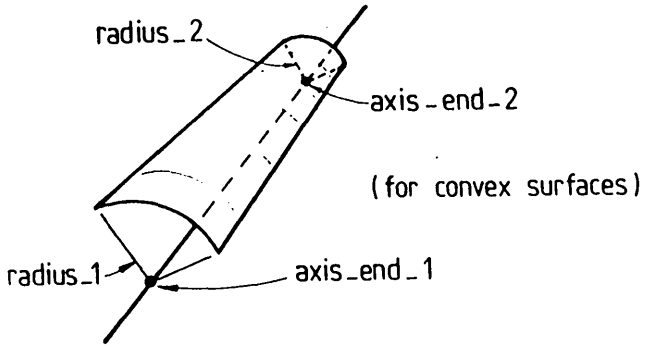


Figure 5–1: Cylinder Surface Definition

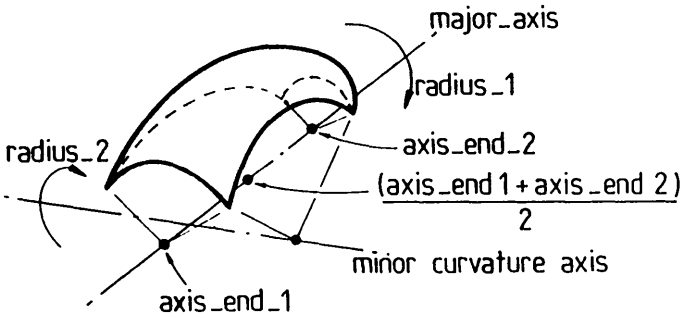


Figure 5–2: Bisurf Surface Definition

views are equivalent. The tangential generator data boundaries approximately match the splitting boundaries of the definition. For surfaces other than this – with either less than 2π or with a deformation that distinguishes orientation, the representation (this chapter) and hypothesis completion (chapter 10) methods need extension.

For a bi-directionally curved surface, the definition is

BISURF[axis_end_1, axis_end_2, radius_1, radius_2]

where the axis endpoints specify the direction of the axis of the strongest curvature (radius_1). The second curvature axis is orthogonal to both this axis and the given normal direction (see below). (Figure 5–2 illustrates the surface definition.) The middle of the surface patch is the midpoint between the axis endpoints. If both radii are positive, the surface is convex and lies behind the midpoint. If both are negative, the surface is concave and lies in front of the midpoint. If one is positive and the other negative, the surface is hyperboloidal, according to the signs of the radii. This surface model assumes that the surface has everywhere the same curvature as that given for the central point, which is clearly not possible for other than spheres, but this is a convenient approximation. The surfaces are drawn assuming the model (r_1 = radius_1, r_2 = radius_2):

$$(sign_{r_1}) * \frac{(x')^2}{r_1^2} + (sign_{r_2}) \frac{(y')^2}{r_2^2} + \frac{(z)^2}{r_z^2} = 1$$

where $r_z = 50$ (an arbitrary choice that should be made an explicit parameter) and (x', y') are in a coordinate system that places x' along the major curvature axis.

The comments above regarding surfaces that subtend more than π apply here too. Another limitation is that surfaces with twist are neither modeled nor analyzed (chapter 10). Figure 5–3 shows a portion of a cylindrical surface with negative curvature that will be used to describe an office chair back. The model definition for this surface is:

CYLINDER[(0.0,0.0,0.0),(0.0,29.0,0.0),-22.5,-22.5]

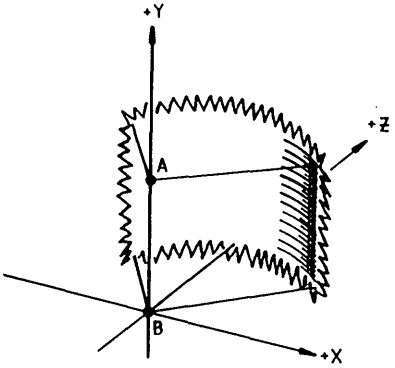


Figure 5–3: Surface Shape for Seat Back (Front Surface)

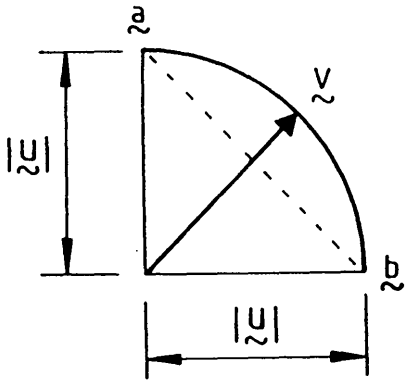


Figure 5–4: Boundary Curve for Given Model Parameters

The extent of a surface is defined by a boundary lying near the curved surface, as approximate but not exact surface patch boundaries are used in the recognition process. The boundary is specified by a few points (in 3D) that lie on the surface and a connecting polycurve: "point – curve – point – curve – ...". The connecting curve descriptions are either straight lines or portions of circular arcs. The surface lies approximately inside the boundary. This implies that surface patches may not smoothly join; this is not a problem, as the models are defined for recognition, not image generation. The arc portions are defined by a direction and magnitude of curvature relative to the endpoints. The arc "$\vec{a}$ – CURVE[$\vec{v}$] – $\vec{b}$" describes the curve passing through $\vec{a}$ and $\vec{b}$ with radius $|\vec{v}|$ and whose center lies on the line passing through the point $(\vec{a} - \vec{b})/2$ with direction $\vec{v}$ (see figure 5–4). Curves with angle greater than π were broken into subsegments to avoid ambiguity over curve definition.

Model surfaces and boundaries are usually segmented according to the criteria discussed in chapter 3 (essentially shape discontinuities). Some exceptions exist. When segmenting curves, points are placed because of difficulties modeling

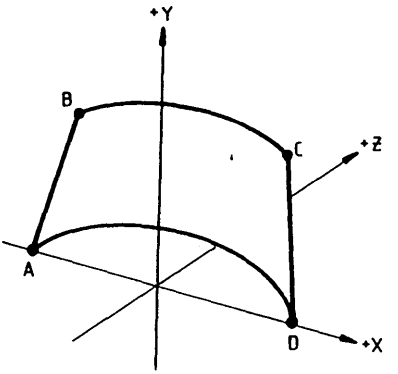


Figure 5–5: Surface Boundary Definition for Seat Back (Front Surface)

arcs subtending angles greater than π, and are ignored during model invocation and matching. Non-segmenting boundaries are used when the surface subtends more than π (in cross section) and are seen as the generators. These boundaries are matched with the front-side-occluding boundaries in the data.

The criterion that caused the segmentation is always recorded as part of the model. The segmentation boundary labels are (refer to chapter 3 for their definition):

BN – non-segmenting generator boundary
BO – surface-orientation
BCW – surface-curvature-magnitude-along-transversal
BCA – surface-curvature-magnitude-across-transversal
BDW – surface-curvature-direction-along-transversal
BDA – surface-curvature-direction-across-transversal

and the segmentation point labels are:

PN – non-segmenting point
PO – boundary-orientation
PC – boundary-curvature-magnitude
PD – boundary-curvature-direction

These labels were not used in the recognition process, but were intended as stronger constraints on the identity of surface groupings.

Figure 5–5 shows the boundary shape for the forward facing portion of the seat back of the office chair model (appendix B). The full definition of the boundary shape is given below.

Each defined surface has a surface normal at a nominal central point. This could be partly calculated from the surface description at the nominal point, but is included here for convenience. All surfaces are presumed to bound solids; laminar surfaces are formed by joining two surfaces back to back. Hence, the normal direction also specifies the outside surface direction. Given the notation described above, the full description of this surface (called "cbackf") is:

```
SURFACE cbackf = PO/(-22.5,0.0,0.0) BO/LINE
                 PO/(-19.5,29.0,10.0) BO/CURVE[0.0,0.0,22.5]
                 PO/(19.5,29.0,10.0) BO/LINE
                 PO/(22.5,0.0,0.0) BO/CURVE[15.91,0.0,15.91]
                 PN/(0.0,0.0,22.5) BO/CURVE[-15.91,0.0,15.91]
                 CYLINDER[(0.0,0.0,0.0),(0.0,29.0,0.0),-22.5,-22.5]
                 NORMAL AT (0.0,14.5,22.5) = (0.0,0.0,-1.0);
```

Figure 5–6 shows the surface and boundary specifications combined to model the chair back.

Objects (called ASSEMBLYs) are described in a subcomponent hierarchy, with objects being composed of either surfaces or recursively defined subobjects. Each ASSEMBLY has a nominal coordinate reference frame relative to which all subcomponents are located. The geometrical relation of a subcomponent to the object is specified by an AT coordinate system transformation that maps from the object's reference frame to the subobject's. This is equivalent to the ACRONYM affixment link. The transformation is specified using an XYZ translation and a rotation-slant-tilt reorientation of the object's coordinate system to the subcomponent's. The transformation is executed in the order: (1) slant the subobject's coordinate system in the tilt direction (relative to the object's XY plane), (2) rotate the system about the object's Z-axis and (3) translate the system to the location given in the object's coordinates. The affixment notation is of the form:

$$((trans_x, trans_y, trans_z), (rotation, slant, tilt))$$

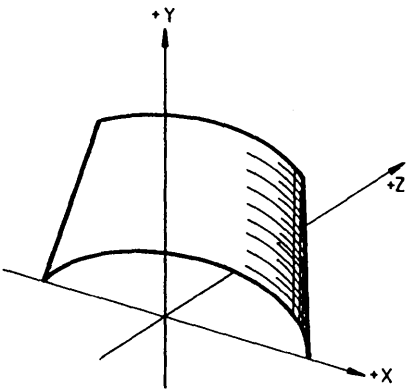


Figure 5–6: Combined Seat Back Model (Front Surface)

Figure 5–7 shows an example of this mapping for the transformation:

$$((10, 20, 30), (0, \pi/2, \pi/2))$$

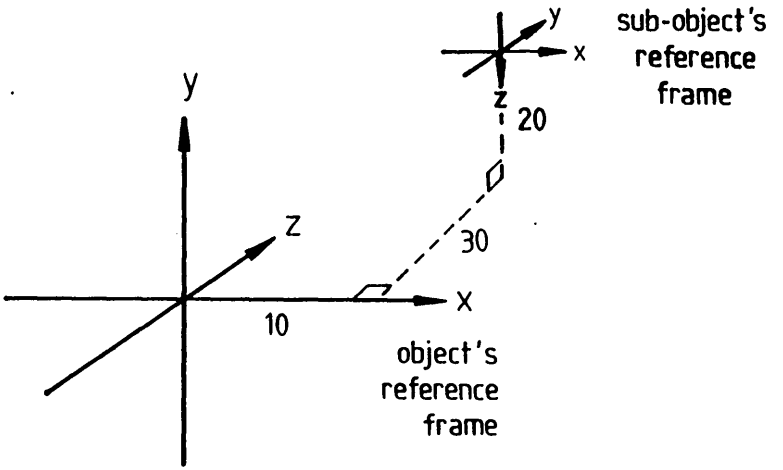


Figure 5–7: Coordinate Reference Frame Transformation

Figure 5–8 shows the robot hand assembly defined from three surfaces: handsidel (the long lozenge shaped flat side), handsides (the short rectangular flat side) and handend (the cylindrical cap at the end). Assuming all three surfaces are initially defined as facing the viewer, the figure specification is:

```
ASSEMBLY hand =
        handsidel AT ((0.0,-4.3,-4.3),(0.0,0.0,0.0))
        handsidel AT ((0.0,4.3,4.3),(0.0,3.14159,π/2))
        handsides AT ((0.0,-4.3,4.3),(0.0,1.5707,3π/2))
        handsides AT ((0.0,4.3,-4.3),(0.0,1.5707,π/2))
        handend AT ((7.7,-4.3,-4.3),(0.0,1.57,0.0))
        ;
```

The defined surface does not completely enclose the assembly, because it was decided that the sixth side would never be seen. This causes no problem; a complete definition is also acceptable as the hypothesis completion process

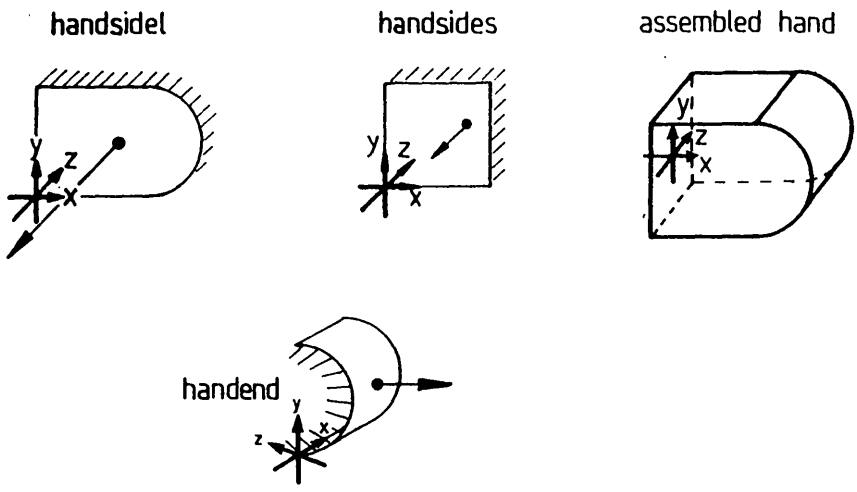


Figure 5-8: Robot Hand Assembly

(chapter 10) would deduce that the surface was not visible when attempting to fully instantiate a hypothesis.

Flexible affixments are specified by using a FLEX (or SYM) option, which allows unspecified translations and rotations of the subobject, *in its local reference frame*, about the affixment point. The flexible attachment definitions use one or more symbolic parameters (as variables). The distinction between the FLEX and SYM option is as follows:

- *FLEX* is used for orientable objects with a flexible affixment between them. The variables in the definition are bound to values when the components are linked during recognition.

- *SYM* is used for unorientable, rotationally symmetric objects. Any value can be matched to the variable during recognition, and the variables are always bound during enquiry (nominally to 0.0). Examples of this from the models used in this research include the chair seat or the chair legs.

The AT and FLEX (or SYM) transformations are largely equivalent, and so could be algebraically combined, but this complicates the definition task. A subobject's flexible position is usually specified in a coordinate system placed relative to its own coordinate system, as the transformations are usually relative to the subobject, not the object. The affixment to the object, however, is usually about a point that is defined in the object's reference frame, so the two transformations are separated. An object S with a rotational degree of freedom is shown in figure 5-9. It is attached to the table T and rotates rigidly (to angle Θ) along the path CD. S and T are part of an assembly Q. Both T and Q have their coordinate frames located at G and S has its at B. The assembly is defined:

```
ASSEMBLY Q =
    T AT ((0,0,0),(0,0,0))                ; T is at G
    S AT ((10,0,0),(3π/2,π/2,0))          ; from G to A
         FLEX ((-7,0,0),(Θ,0,0))          ; from A to B
    ;
```

The recursive subcomponent hierarchy with local reference frames supports a simple scheme for coordinate calculations. Assume the hierarchy of components:

```
ASSEMBLY P0 =
                P1 AT A1
                FLEX F1
ASSEMBLY P1 =
                P2 AT A2
                FLEX F2
ASSEMBLY P2 =
                P3 AT A3
                FLEX F3
(etc)
```

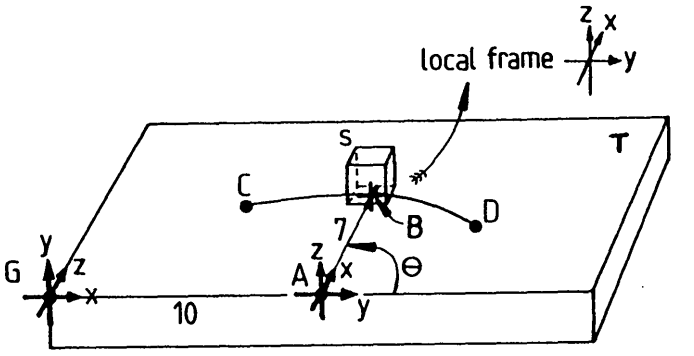


Figure 5–9: Flexible Assembly Example

Let A_i and F_i be simplified homogeneous coordinate matrices representing the reference frame maps from above. Then, each of these 4*4 matrices has the form:

$$\begin{pmatrix} R_i & \vec{T}_i \\ \vec{0}^t & 1 \end{pmatrix}$$

where R_i is the 3×3 rotation matrix and $\vec{T}_i$ is the translation vector. Each matrix represents the mapping $\vec{T}_i + R_i * \vec{v}$ of a vector $\vec{v}$ in the subobject to the object coordinate system. If:

G is the matrix mapping from the object's top level coordinate system into global coordinates, and

C maps from global coordinates into those of the camera,

then a point $\vec{v}$ in the local coordinate system of assembly P_n can be expressed in camera coordinates by the calculation:

$$CG(A_1F_1^{-1})(A_2F_2^{-1})...(A_nF_n^{-1})\vec{v}$$

There are many ways to decompose a body into substructures, and this leads to the question of what constitutes a good segmentation. In theory, no hierarchy is needed for rigidly connected objects, because all surfaces could be directly expressed in the top level object's coordinate system. This is neither efficient (e.g. may represent repeated structure) nor captures our notions of substructure. Further, a chain of (more than 2) flexibly connected subobjects represented in a single reference frame would have a complicated linkage definition. Some guidelines for the decomposition process are:

1. Object surfaces are segmented according to the shape discontinuity criteria of chapter 3.

2. Flexibly connected substructures are distinct ASSEMBLYs.

3. Repeated structure forms distinct ASSEMBLYs (e.g. a common surface shape or subassembly, like a chair leg).

4. Surface groups surrounded by concave surface shape discontinuities are units (e.g. where the nose joins to the face). This is because one cannot distinguish between a connecting-to or adjacent-to relationship, and so data segmentation must take the conservative choice. Hence, the models should follow this as well. Figure 5–10 illustrates this problem for a rivet versus a cylinder on a plane.

5. Objects commonly named distinctly *might* be modeled distinctly. (There is probably some epistemological principle involved here.)

Performance

The full geometric models and drawings of the objects used in this research are shown in appendix B. A reasonable range of solid and laminar structures are modeled. The robot example demonstrates the use of the FLEX option to join flexibly connected components. The trash can and chair illustrate laminar surfaces and symmetric subcomponents (SYM). All the objects together illustrate

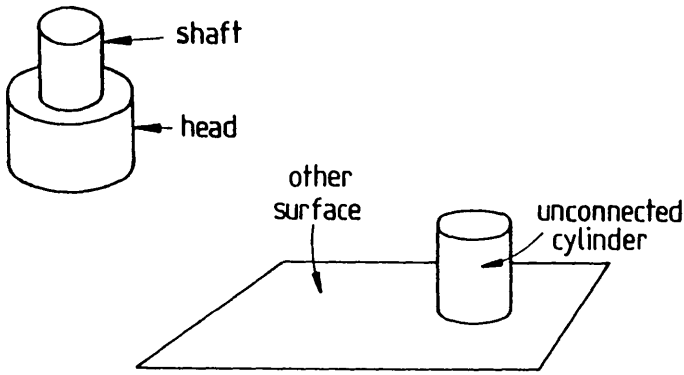


Figure 5–10: Segmenting a Rivet Versus a Cylinder on a Plane

most of the segmentation types, except for both curvature direction discontinuities, and curvature magnitude discontinuities occurring across a surface path.

Criticisms

There are several major inadequacies with the geometric modeling system. Object dimensions have to be fixed, but this was largely because no constraint maintenance mechanism was available, and this aspect was not generally relevant to the research reported in this thesis. Minor object variation could be added by specifying the coordinate system transformations using a range of parameters, rather than with a constant map. This would support variation in feature placement, but not feature size; however, the key factors in surface description are class, curvature and relative placement, and size is secondary. For full definition of an object by its shape, but not its size, the translational component of surface placement would also need to be parameterized.

Uniform surface and boundary segment curvature was also a simplification.

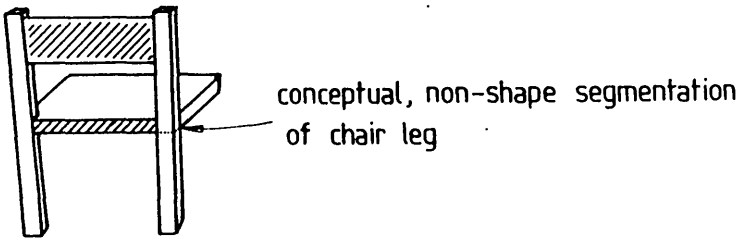


Figure 5–11: Chair Leg Becomes Part Of Chair Back

However, because major curvature changes cause segmentations, deviations between the models and objects should be minor. The point was to segment the models to correspond with the data, in order to promote direct feature matching through having the same representation for both. A more exact metric surface representation may be desirable for representing finer details, though they may just appear by using a finer scale of segmentation.

Some surfaces do not segment into conceptual units strictly on shape discontinuities, such as when a chair leg continues upward to form part of the chair back (figure 5–11). Here, the segmentation requires a boundary that is not in the data. This is part of the more general problem segmentation problem which is ignored here.

Another problem with modeling was the introduction of non-segmenting boundaries. These usually occur on symmetric objects, so match with occluding boundaries. However, if the two symmetric halves of a surface are explicitly modeled, then duplicate recognitions occur (chapter 10). A similar problem is with the robot shoulder body (robshldbd, see appendix B), which had its surface

split into two halves even though it is not symmetric. So, to be matchable, the shoulder body can only be seen from the side where the occluding boundary will correspond to the segmentation boundary . These are deficiencies of (a) the modeler and (b) hypothesis construction, which (as yet) cannot reason about surfaces that wrap around an object.

Further, the surfaces have been represented with a single boundary, and so must not have any holes. The important issues of scale, natural variation, surface texture and object elasticity/flexibility are also ignored, but this has been true of almost all modeling systems.

This surface-based representation method seems best for objects that are primarily characterized as constructed solids. Hence, many objects, especially natural ones, would not be well represented. The individual variation in a tree would not be characterizable, except through a surface smoothing process that characterized the entire bough as a distinct solid with a smooth surface, over the class of all trees of the same type. This is an appropriate generalization at a higher level of conceptual scale. Perhaps a combination of this smoothing with Pentland's fractal-based representation of natural texture ([PEN83]) could solve this problem.

An object probably should not be completely modeled – only the key features need description, according to the goals of the recognition system. In particular, only the larger surfaces and features need representation and these should be enough for initial identification and orientation. While a shoe could be exactly represented using surfaces at some scale, this is probably not the best representation for recognition. Intuition suggests that its model should have a toe region, a foot hole, an general wedge shape, a sole and heel and a lace region.

This suggests that some generalization of structure would probably be appropriate for more general modeling, and that matching should probably be based more in the suggestive direction away from pure metrics. This thesis makes a small step in this direction by making the primary characteristics of an object be distinct surface patches (defined by curvature class, curvature parameters,

orientation and nominal placement), rather than a exact metrical description of the object.

Other Extensions

Because the segmentation goal is to produce similar segmentations for the models and data, it should be possible to automate the construction of the geometrical body model (at least at a single level of scale). This is ultimately necessary because of the impossibility of manually constructing models for thousands of real objects. For rigid objects, the surface and boundary parameters from the data would become those of the model, and their feature relationships fix the coordinate transformations relative to the whole object. Popplestone et al ([POP75]) investigated this problem for simpler surface classes on rigid bodies, using data obtained from a striper. Potmesil ([POT83]) has made a start at this problem for uniformly spline-represented surfaces. Difficulties that need solution are:

- completing the full 3D model from several views.
- deducing which objects are flexibly connected and what are the directions of flexibility.
- deducing the symmetry of objects (Brady and Asada [BRA84b] have investigated smoothed local symmetries).
- segregating distinct subobjects and unifying identical instances of these.

Finally, any realistic object recognition system must use a variety of representations, and so the surface representation here should be augmented. Several researchers ([NEV77],[BRO81],[MAR82]) have shown axes of elongated regions or volumes are useful features, and volumetric models are also useful for recognition. Reflectance, gloss and texture are also good surface properties. Viewer centered and sketch models provide alternative representations.

5.3 Other Object Information

Geometrical models represent most of the information needed for recognition. This section discusses the remaining information needed for the research and its representation.

Invocation (chapter 9) needs three types of object-oriented information. The first is inter-object relationships, which provide indirect evidence. There are five types of relationships between objects: subcomponent, supercomponent, subtype (specialization), supertype (generalization) and arbitrary association. A weight is needed to express the importance of the relationship. The information is represented as:

(relation, objecttype1, objecttype2, weight)

The second requirement is for direct evidence constraints. They specify the acceptable value ranges on different attributes (chapter 8), and a contribution weight for each attribute. Examples of this information are the expected areas of surfaces or angles at which surfaces meet. The information is represented as:

(object, attribute, low value, high value, weight)

The final invocation requirement is for subcomponent groups, which specify those features of an object likely to be seen together from particular viewpoints.

Some of the extra information could have been derived from the geometrical models (e.g. subcomponent relationships). For others, like the importance of an attribute towards invocation or the relationship between the object's shape and these properties is not well understood. Invocation needs information based on the associations between objects, rather than their structure. Hence, some extra-geometric information is represented.

This chapter presented an object representation based on surface shape. The representation consisted of:

- shape segmentable surface patch primitives defined by their shape and extent,

- assemblies defined by hierarchical placement of subcomponents in a local reference frame, with use of variables for flexible connected subcomponents, and

- object associations and properties as needed for model invocation.

The chief contribution of the chapter was in the definition of a surface oriented model designed for recognition, rather than image generation.

Chapter 6

Making Complete Surface Hypotheses

The first step in interpreting surface information is the formation of surface hypotheses. This process produces symbolic entities that relate surface image regions to specific patches of as-yet-unidentified object surfaces. This is the important first transformation of image-based data to object-based data. Further, it reduces data complexity by representing a surface region by a single data structure.

Some cases of partial occlusion can be corrected during surface hypothesis formation. Here, the observed surface is a subset of the hypothesized object surface with the missing portions of obscured surfaces reconstructed. This is useful for helping recognition continue even if the object is partially obscured.

This chapter will show by using a labeled, segmented surface image:

- that the transformation from image regions to object surface hypotheses is simple, and
- that the most common cases of occlusion can be overcome with the surfaces largely reconstructed.

6.1 Making Complete Surface Hypotheses

This process starts with the region graph as described in chapter 3. From this, a set of hypotheses about the object surfaces is extracted to produce the first explicit object-oriented representation.

Initially, the surface hypotheses are identical to the image regions, except for the conceptual association of the surface patch with a part of an (unidentified) object. Then, other surface hypotheses are created based on surface extensions consistent with presumed occlusions. While it is obviously impossible to always correctly reconstruct obscured structure, in many cases a single surface hypothesis can be created that joins consistent visible surface parts. Figure 6–1 shows a simple case of surface reconstruction.

In figure 6–2, four cases of a singly obscured surface are shown, along with the most reasonable reconstructions possible. In the first case, the original surface boundaries meet when they are extended, and this is presumed to reconstruct a portion of the surface. If the boundaries change their curvature or direction, then reconstruction may not be possible, or it may be erroneous. (Even if erroneous, the reconstructed surface may more closely approximate the true surface than the original input.) The second case illustrates when reconstruction does not occur, because the unobscured boundaries do not intersect when extended. The third case shows an interior obscuring object removed. The fourth case shows where the surface has been split by an obscuring object and reconstructed. The boundary extension always remains back-side-obscuring, because it might participate in further reconstructions.

What is interesting is that only three rules are needed for the reconstruction (see figure 6–3). They are:

1. remove interior closer surfaces (part a),

2. extend and connect boundaries on separate surface hypotheses to form a single surface hypothesis (part b), and

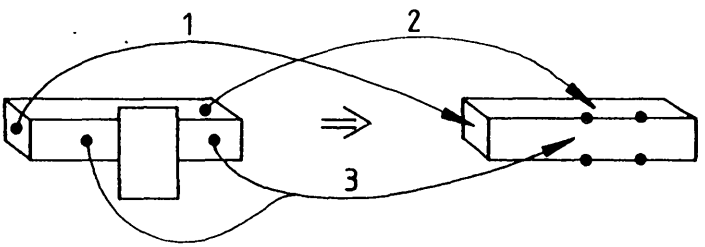


Figure 6–1: Surface Hypothesis Construction Process

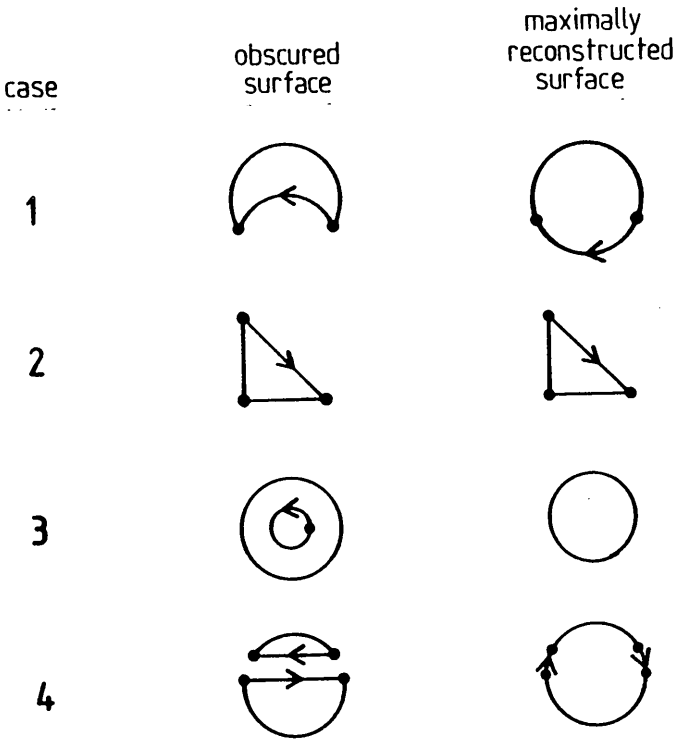


Figure 6–2: Four Occlusion Cases Reconstructed

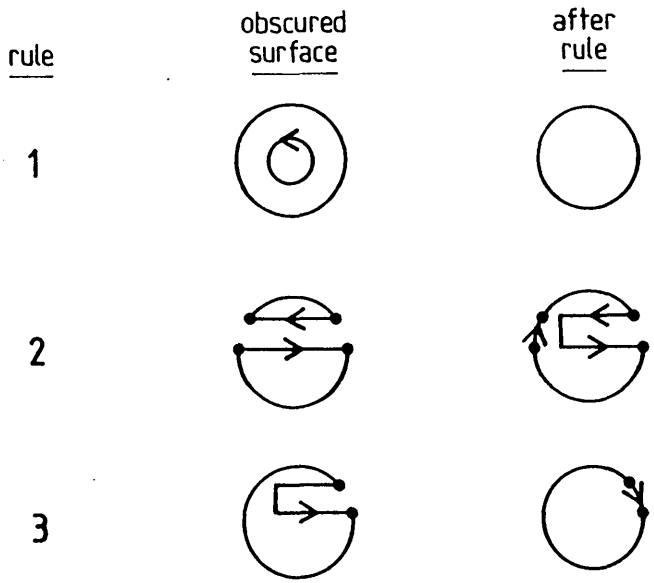


Figure 6–3: Surface Completion Processes

3. extend and connect boundaries on a single surface to make a larger surface (part c).

Rule 2 connects two separated surfaces if either extension of the boundaries intersect. The remaining portion of the obscuring boundary is disconnected to indicate no information about the obscured portion of the surface (until rule 3). Rule 3 removes notches in surfaces by trying to find intersecting extensions of the two sides of the notch. Repeated application of these rules may be needed to maximally reconstruct the surface.

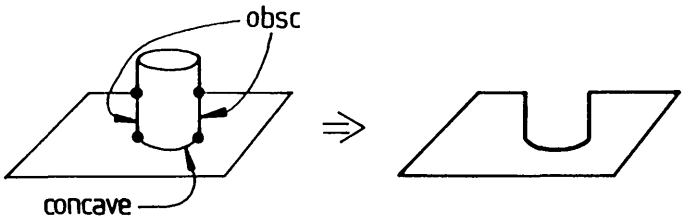


Figure 6–4: Concave Boundaries Also Delineate Obscured Regions

The criteria for when these reconstruction rules are applied and details of their application are now given.

Reconstruction is attempted whenever surface occlusion is detected, which is indicated by the presence of back-side-obscuring boundaries. Concave orientation discontinuity boundaries also imply potential occlusion. In figure 6–4, the base of the obscuring cylinder rests on the plane and so has a concave shape boundary, which should be treated as part of the delineation of the obscured region of the plane. (If the two objects were separated slightly, an obscuring boundary would replace the concave boundary.) As the concave boundary does not indicate which object is in front, it is assumed that either possibility could occur.

It may not always be possible to determine when concave boundaries should be considered in this manner. Figure 6–5 shows a concave boundary that is not usually associated with obscured structure, though there could be some in a situation where there are perfectly aligned surfaces.

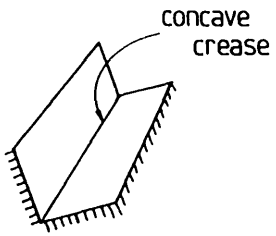


Figure 6–5: Concave Boundaries Don't Always Imply Reconstruction

Figure 6–6 shows the more usual case, where one surface sitting on another will create a tee junction, or connect to an obscuring boundary (as in figure 6–8).

To summarize, occluded surface portions are implied by back-side obscuring and concave shape discontinuity that either form a closed loop or end as part of the crossbar of a TEE junction.

After finding the segments indicating where reconstruction is needed, the points where reconstruction starts need to be found. These are the ends of boundary segments that meet these criteria:

1. the endpoints must lie between obscured section boundaries (defined above), and boundaries that definitely lie on the object surface (i.e. convex or front-side-obscuring).

2. The segments must join at a TEE junction.

3. The true object surface boundary must be the shaft of the TEE.

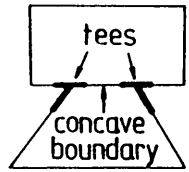


Figure 6–6: Tee Junctions Delimit Reconstruction Concave Boundaries

These points are illustrated in figure 6–7.

To reconstruct, boundary segments must be extended and must intersect. As intersecting 3D surface curves still intersect when projected onto the image plane, extension and intersection is done only in 2D, thus avoiding the problems of 3D curve intersection. Extending the boundaries is done by estimating the curvature shortly before the terminating tee and projecting the curve through the TEE. By the boundary segmentation assumptions (chapter 3), segments could be assumed to have nearly constant curvature, so the extension process is justified.

These analyses have shown what needs to be done for reconstruction, what is replaced and where it takes place. Before reconstruction is allowed, other constraints must be satisfied:

- If a portion of a surface is obscured, then that portion must be completely bounded by an appropriate type of boundary (as defined above).

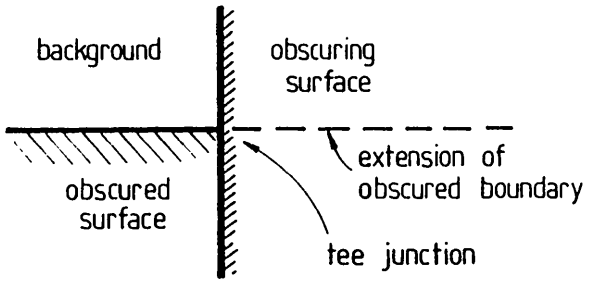


Figure 6–7: Reconstruction Starts at a TEE Junction

- The ends of the unoccluded portions of the surface boundary must be joinable.

- The joined portions of surface must lie on the same side of the boundary extension.

- The obscured portions of a surface's boundary can not intersect other boundaries of the same surface. (This rules out obscure laminar surface reconstructions, where the surface may cross underneath itself.)

- Surface fragments being reconnected must have consistent depths and surface orientations. (This is valid because surface shape segmentation enforces the shape consistency.)

- Two reconnected surfaces must not be otherwise adjacent.

There are many partially obscured surfaces in a typical image and simple extension of tee junctions will cause many intersections and potential recon-

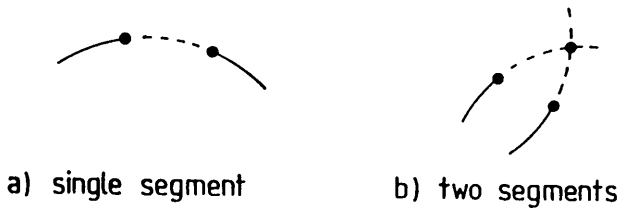


Figure 6–8: Segment Extension Process

structions. Fortunately, these other constraints rule out most inappropriate reconstructions.

There are two outputs from the reconstruction process – the boundaries and the shape of the surface. Because of the surface segmentation assumption, the reconstructed surface shape is an extension of the visible surface's shape. The boundary problem is different because, in the absence of any knowledge of the object, it is impossible to know exactly where the boundary lies. It is assumed that the obscured portions of the boundary is an extension of the unobscured boundaries, and continues with the same shape until intersection. The two cases are shown in figure 6–8. In case a, a single segment extension connects the boundary endpoints with the same curvature as the unobscured portions. In case b, the unobscured portions are extended until intersection.

The above theory defines the surface reconstruction process for correcting individual occlusions. This process is applied until maximal surfaces are created. Figure 6–9 shows an example of this.

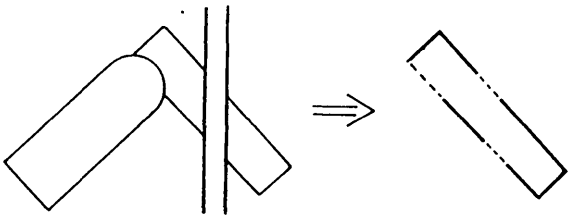


Figure 6–9: Multiply Obscured Surface Extended

The process may produce spurious hypotheses from coincidental alignments. Hence, the conservative approach to producing surface hypotheses would be to allow all possible surface reconstructions, including the original surface without any extensions. This proliferates surface hypotheses causing combinatorial problems in the later stages. So, the final surface hypotheses are made from only the maximally reconstructed surfaces. If the reconstructed surface is larger than the true surface, invocation may be degraded, but hypothesis completion would continue because the surface extents are not used. Verification avoids this problem by using the original image regions as its input. Because of the constraints of boundary intersection and labeling and surface shape compatibility, few spurious reconstructions are likely to occur.

The extension processes are obviously not perfect, as seen in figure 6–10. In the first case, the extended segments never intersect, and in the second, extension creates a larger, incorrect surface. These problems cannot be avoided without more detailed reasoning. As the goal was to reconstruct enough to allow the rest of recognition to proceed, a few failures should be acceptable.

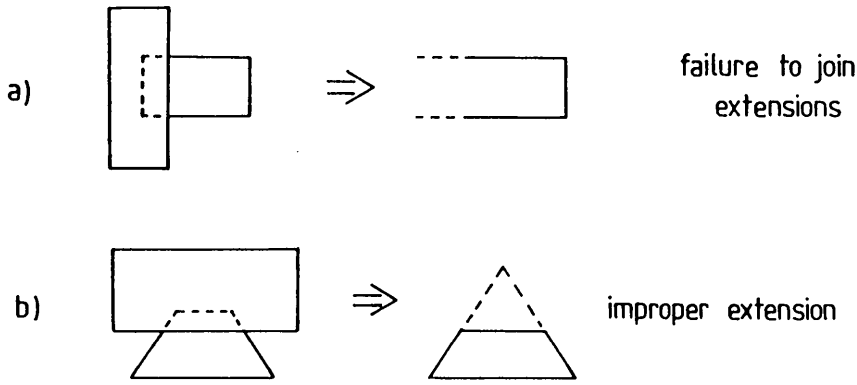


Figure 6–10: Unsuccessful Extensions

The Surface Hypothesis Graph

The region graph (chapter 3) forms the initial input into the explicit surface hypothesis process. The surface hypothesizing process described in this section makes the following additions to the graph:

1. Every surface hypothesis node links to a set of region nodes.

2. A surface node is linked to a chain of boundary nodes linking to boundary segments that isolate the surface.

3. If two surface nodes have region nodes linked by adjacency nodes, then the surface nodes are linked by adjacency nodes.

6.2 Evaluation: Making Complete Surface Hypotheses

This section evaluates the theory of creating explicit surface hypotheses, as presented in the previous section. The three topics are evaluation criteria, performance, and critical discussion.

Evaluation Criteria

Because of the pervasiveness of occlusion in natural scenes, rough surface reconstruction is necessary. Reconstructed surfaces can only be hypothetical but, because of the surface segmentation assumptions (chapter 3), there are no extreme surface or boundary shape variations in a single segment. As many natural object boundaries exhibit continuity over moderate distances (at appropriate scales), reconstruction should be possible. This is even more likely with most man-made objects. Hence, boundary extrapolation and surface completion can overcome the most common cases of occlusion: internal closer objects, partial surface overlap, and split surfaces.

To show this, several examples of program performance will be given. This will show that both problem constraints are generally reasonable and the implemented theory produces the desired results.

Appendix A has the two test images used throughout this thesis. Figures A–6 and A–15 show the initial image regions. Figures 6–11 and 6–12 show the final surface hypotheses formed (numbers in the picture are the surface index).

There are several instances of successful surface reconstructions in the test images. Figure 6–13 shows reconstruction of the robot upperarm side panel from test image 1. The surface merging operation has rejoined the two sections, and the boundary extension has largely restored the missing section in the lower left corner. Because the reconstruction is only suggestive, and because the leftmost

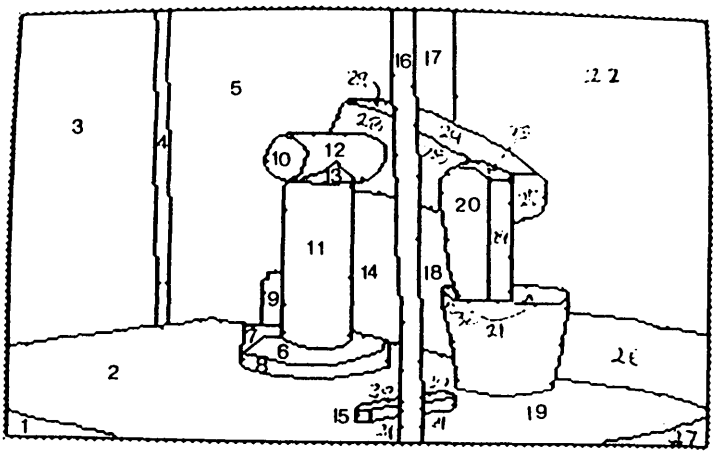


Figure 6–11: Surface Hypotheses for Test Image 1

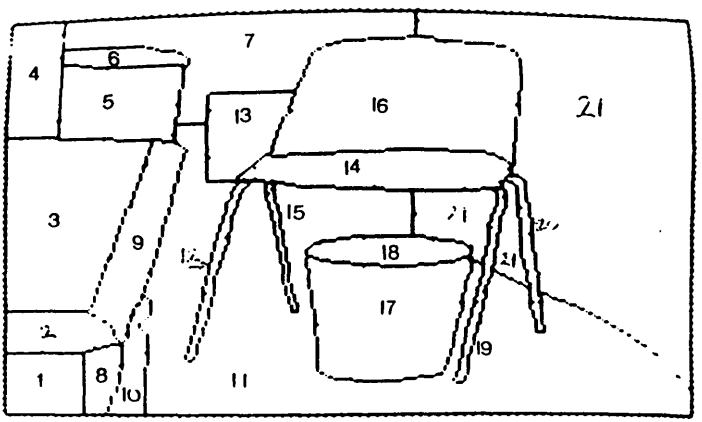


Figure 6–12: Surface Hypotheses for Test Image 2

boundary was not observed as being curved, the reconstruction is a little generous and slightly exceeds the actual surface. The right end of the panel could not be reconstructed because of insufficient evidence for the true boundary, although the labeling claimed it was obscured. Other valid reconstructions included the block lying on the table and the two halves at the back of the trashcan. The latter is interesting because it was a curved surface, so matching of surface shape had to account for the change in orientation. One inappropriate merging occurred: the two top panels of the robot upperarm had their real join hidden behind the vertical column. As they were continuous, their orientation differed only slightly, and met all the occlusion constraints, they were merged as well. The hypothesis construction process expects this type of error, so it did not prove catastrophic. Another minor error occurred: the two table surfaces (regions 5 and 6) were not connected because of an error in extrapolating surface depths.

Figure 6–14 shows the reconstruction of the fragmented back panel from test image 2. Here, several applications of the reconstruction rules were needed to piece together the panel and reconstruct the missing portions.

These examples show that the occluded surface reconstructions are successful, and figures 6–11 and 6–12 show that most reasonable surfaces are made.

Criticisms

The major problem with the surface reconstruction constraints is unsolvable – one cannot reconstruct the invisible when the image is taken from a single viewpoint. Stereo or observer movement would help reconstruct the surface, however, and the occlusion cues could tell the observer when to do it.

Another obvious criticism is over performance when applied to rich natural images. The criteria are successful surface reconstruction, few spurious hypotheses, and no combinatorial explosion of legitimate hypotheses. Realistic images are likely to have missing or erroneous data, such as for line labels or surface orientation. These problems will degrade both the quality and rate of performance, though it is hard to predict how much. In short, the processes described

Figure 6–13: Upper Arm Surface Reconstruction from Test Image 1

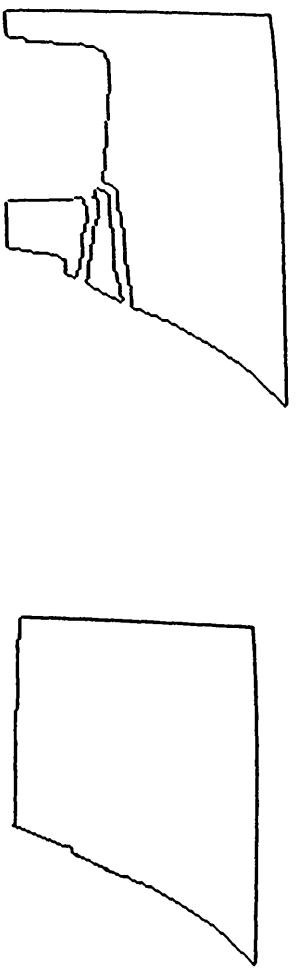


Figure 6–14: Back Panel Surface Reconstruction from Test Image 2

in this chapter seems fine for clear laboratory images, but it is hard to predict their performance on natural scenes.

Extensions

To avoid the problem of redundant hypothesis formation, only the reconstructed surface is kept. A better solution might isolate the description of the surface, which is common to all hypotheses, from that of the boundary of the surface. A similar problem occurs with joining two separated surfaces. This produces fewer side effects, as the reconstructed surface is unlikely to be interchangeable with either of its visible portions. An error for this case occurred when regions 17 and 32 were merged in test scene 1.

The second extension is more fundamental. Because segmentation criteria may break up or merge surfaces at different scales, surface hypotheses need to allow for alternative representations derived as a function of a locally relevant scale. These representations are related but are not interchangeable, nor are they equivalent. One problem with scale is how to reconstruct the structures derived at different scales in a way that facilitates later processing and does not lead to combinatorial explosion.

Scale also affects surface reconstruction, as can be seen in figure 6-15. The first figure shows the extension based on individual teeth of the gear, whereas at the larger scale, the extension is based on the smoothed convex envelope of the gears.

Contributions

This chapter:

- showed that by using a labeled, segmented surface image object surface hypotheses were readily formed, and

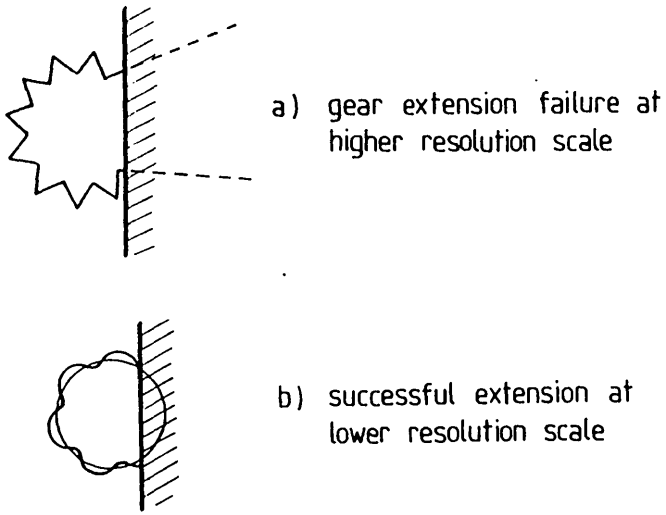


Figure 6–15: Scale Based Extension Problems

- **presented rules and constraints for reconstructing surfaces affected by the most common types of occlusion.**

Chapter 7

Surface Clusters

A competent object recognition system clearly needs a figure/ground separation mechanism, to indicate both which image features are related and the object's spatial extent. As surface information is explicit in the image data and as the surface is the physical delimiter between object and non-object, surfaces can provide the basis for segregation. This chapter identifies the need for a new intermediate representation (surface clusters) between the $2\frac{1}{2}$D sketch and the model based 3D object hypotheses. The clusters segment the image into blob level, identity independent solids. Rules for producing surface clusters are elaborated and evaluated.

The goal of the "surface clusters" formation process is to group the segmented surface image regions that belong to the same object. These clusters form the first object-level interpretation of an image, by the transformation from two dimensional to three dimensional scene understanding. Surface clusters capture the intuitive notion that: "There is a potential distinct 3D object with surface shape X".

Figure 7–1 shows a typical scene and figure 7–2 shows some of the corresponding surface clusters.

This chapter has three goals: to motivate the use of the grouping process, to present its theory, and to evaluate the process.

Figure 7–1: Intensity Image With Surface Region Boundaries

Figure 7–2: Surface Clusters

7.1 Why Surface Clusters?

A surface cluster is a maximal set of surface hypotheses such that any one surface is adjacent to at least one other within the set with a suitable connecting boundary between the two. That is, a surface cluster tries to recreate the complete, visible, 3D portion of each distinct object's surface.

There are three motivations behind the creation of surface clusters:

1. the creation of a blob-level object representation,

2. the reduction of search complexity through structuring image features, and

3. the focusing of attention on an object and its associated image features.

The first motivation for a surface cluster is a competence issue – such an aggregation is a new conceptual element. It is an "unidentified, but distinct" object interpretation associated with clusters of image features. This is a "blob" level representation describing solid objects with approximate spatial relationships but without identifications, which is useful for some tasks such as navigation, object avoidance or object interception. It helps bridge the conceptual distance between the object and the image, so it is an important visual representation. With this structure, the key image understanding representations now become: image – primal sketch – surface image – surface clusters – objects

This level is important because such interpretations are needed for unidentifiable objects, whether because of faults or lack of models in the database. It adds an element of robustness to the total theory, as some intermediate levels of interpretation may be achieved even when full identification is not. The grouping also creates a good starting point for further interpretation; it is a figure/ground separation for solid objects. The rudimentary object has properties which are used to invoke interpretations (as will be seen in Chapter 9).

Data objects should be defined using properties that are easily extractable, thus making the data to symbol transformation process both simple and robust.

The surface cluster aims at grouping surfaces into object-based but identity-independent structures, which will be shown to be an easy transformation.

The second motivation for creating these aggregations is one of performance. By isolating those surfaces that are interrelated, an immediate reduction in the complexity of the analysis results. The whole interpretation has been reduced to a set of smaller independent problems, which is necessary given the quantity of data in an image. A casual mathematical analysis supports the performance argument. Since every surface on an object has a relationship to every other surface on the object, an object with N visible surfaces has $O(N^2)$ relationships. If there are A objects in the scene, each with B visible surfaces, there will be AB total visible surfaces. So, initially, the analysis problem is $O((AB)^2)$. If we partition the image into the A objects, the analysis problem is then $O(AB^2)$. For reasonable scenes $A = 50$ is a nominal value, so the aggregation process leads to a remarkable improvement in performance.

There is also a more intuitive motivation, in that the aggregation process has a focusing effect. For certain types of information, it partitions the information into relevant groups. (However, there are many types of information, and some will come from external objects that may be related to the current object at a higher level, as in separate wheels on the same automobile.) Here, the only information affected is that concerned with inter-surface relationships (i.e. relative surface orientation will not be considered across segmentation boundaries, nor will unrelated surfaces be matched to the same model instance). Thus, the process creates activity contexts for the later stages of recognition.

Surface clusters will not be perfect. They may be incomplete, as when an object is split up by a closer obscuring object, though the surface hypothesizing may bridge the occlusion. They may also be over-aggregated – from images where there is insufficient evidence to segregate two objects. The goal of the process is to produce a partitioning without a loss of information. These failures may reduce recognition performance (i.e. speed), but not its competence (i.e. success): incompletely merged surface clusters will be merged in a larger context

(section 7.2) and insufficiently split surface clusters will just cause more searching during hypothesis completion.

As discussed in chapter 2, previous research showed how to create rough polyhedral object grouping in images using line labeling cues. There, isolating boundaries were determined by connected chains of separable concave and obscuring boundaries in polyhedral domains. Here, surface shape boundaries that connect surfaces are found, and the transitive closure of the connection relationship gives the clusters. This allows application to objects with curved surfaces and some laminar surface groupings. In particular, obscuring boundaries can become connecting (in certain circumstances) which allows the two laminar surfaces in the folded leaf problem to become joined into a single surface cluster (figure 7–5). Further, concave boundaries define ambiguous depth relationships between primitive surface clusters. This can be exploited to limit combinatorial explosion in the creation of larger surface clusters, which is necessary to provide the image context for structured object recognition.

7.2 Theory: Surface Clusters

This process collects the completed surface hypotheses (chapter 6) to produce the visible surface for the object(s). Though the surface is incomplete (i.e. missing the back sides), it is a solid bounded in front by the visible surfaces, and forms the first 3D representation for the whole object.

The two important issues are: what are significant data objects and how is image connectivity determined. The first issue is concerned with defining what types of image structures should be considered distinct and the second focuses on the criteria for cluster membership.

The goal of the process is to produce clusters of image surfaces that correspond to distinct model structures, through grouping image surfaces into minimal isolated subcomponents. The solution proposed is conservative, in the sense that it avoids splitting these minimal subcomponents at the expense of merging

distinct adjacent objects. The effect of this is to produce contexts guaranteed to contain complete primitive assemblies (as defined in chapter 5), but may contain more than one. (Though the fewer the more successful segmentation has been.) Primitive surface clusters are those that cannot be split further and larger surface clusters are formed from groups of primitives.

Splitting a real object between several different surface clusters would be catastrophic because it asserts that the segregated components are unrelated. Creating blobs larger than single objects is mainly an annoyance, because the rest of the recognition process should eventually pick out the objects in the surface cluster.

Determining Segmentation Boundaries

The key to the significant data object definition problem is boundary type. It was assumed that the modeled objects have been segmented into subcomponents at chains of connected concave boundaries (see chapter 5). As concave model boundaries form concave image boundaries, the latter are potentially segmenting. In figure 7–3 part a, the potential segmenting boundary is truly segmenting.

Concave surface orientation boundaries allow separate objects to be on opposite sides of the boundary. For example, a block sitting on a table has concave boundaries isolating it from the table. Nameable subcomponents of natural objects often fit flushly with concave boundaries, as in the nose-to-face junction. Because the boundary is concave, it is indeterminate whether the two surfaces are joined or merely contact. So, the conservative approach suggests that this boundary is provisionally segmenting. Assuming concave boundaries always imply segmentation leads to contradictions as seen in figure 7–3 part b, where there is no reasonable shape boundary at point X to continue segmentation. If there are other paths between the two surfaces that do not cross a segmenting boundary, then the final segmentation will not include this boundary.

Whenever one object sits in front of or on top of another, the intervening boundary is always either concave or obscuring, as illustrated in figure 7–4. To

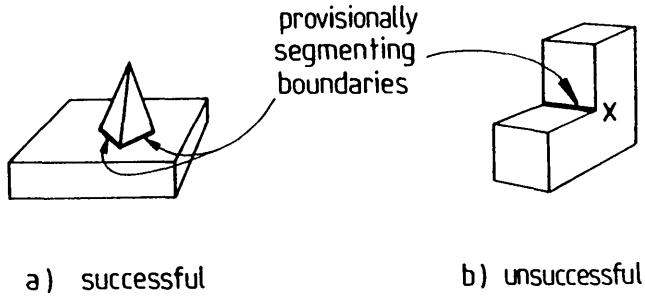


Figure 7–3: Concave Boundaries Provisionally Segment

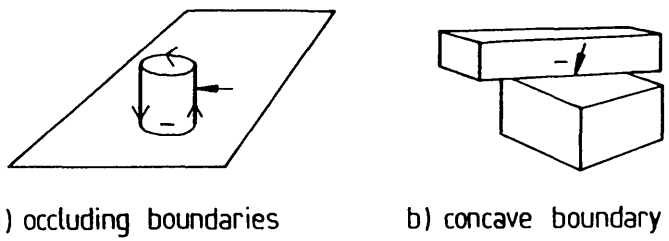


Figure 7–4: Object Ordering Causes Concave and Obscuring Boundaries

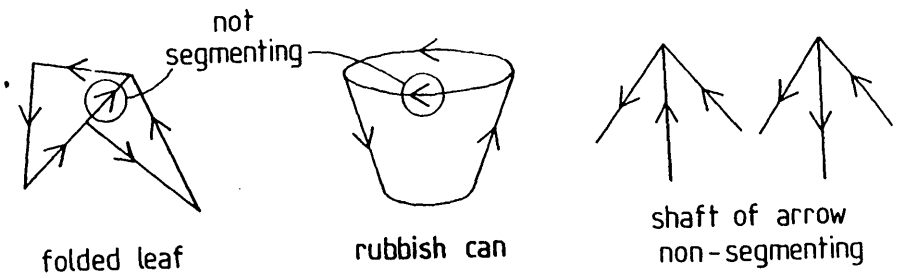


Figure 7–5: Connectivity Holds Across Some Obscuring Boundaries

complete the isolation of the cylinder in part a from the background plane, a rule is needed to handle occluding boundaries. As these usually give no cues to the relation between opposing surfaces (other than being depth separated), surfaces will usually be segmented across these.

Connectivity sometimes holds across some obscuring boundaries. Disregarding coincidental alignments, the one exception found occurs when two-dimensional objects fold back on themselves, as illustrated in figure 7–5. This figure shows a leaf folded over and the two surfaces of a trashcan. In both cases, the two surfaces are connected, even though an obscuring boundary intervenes. Fortunately, this case has a distinctive signature: the arrow vertex shown at the right side of the figure. Viewpoint analysis (by octant) shows that this is the only special trihedral label case needed for two connected laminar surfaces.

If two surfaces lying on opposite sides of convex boundaries belong to two different objects, then the two objects are coincidentally aligned. Hence, it is assumed that the two surfaces ordinarily belong to the same object. A surface

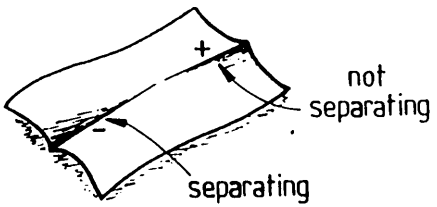


Figure 7–6: Separation Does Not Always Propagate Along Boundaries

orientation boundary that is both concave and convex in places is broken up by the segmentation assumptions.

Given the current input data, it is possible to directly infer from surface orientation whether the surface junction is concave (e.g. [SUG79]). If the orientation data were missing, then topological analysis like that from the blocks world analysis (e.g. [WAL75]) can sometimes uniquely deduce that a particular boundary is concave. Labeling rules could also correct some data errors (e.g. [FAL72]).

Single boundary segments are now labeled as "segmenting" or "non-segmenting". Unfortunately, "segmenting" does not always propagate along boundaries. Figure 7–6 part a shows where it does and part b shows a counterexample, where two curved surfaces change their relative orientation along their common boundary so a concave segmenting boundary becomes a convex non-segmenting boundary. Hence, until further constraints are discovered, only the boundaries so far labeled as segmenting will be treated as such.

To summarize, the constraints that specify the segregation of surface regions are:

- Connectivity does not hold across concave shape boundaries (figure 7–4).
- Connectivity does not hold across obscuring boundaries, except when the boundaries are configured as in figure 7–5.

Grouping Connected Surfaces

The key to the surface cluster formation process is connectivity. The goal is to associate surfaces that are related by virtue of being connected. In practice, the complementary computation is easier: identify the boundaries that do not connect. Then, surface clusters can be formed by collecting all surfaces regions that have direct or transitive connection.

This reverse formulation also arises because of matching requirements. Surface clusters need to be maximally connected to provide complete contexts for object subcomponents, so if there is doubt then connect. This contrasts with model segmentation, where, if there is doubt then segment. Following this, all model features will lie within some data context.

The constraints that specify the construction of the primitive surface clusters are:

- Adjacent regions whose common boundaries are not all segmenting are connected.
- If two regions are each connected to a third region, then the first two are connected to each other.

These constraints specify the computation that aggregates surfaces into primitive surface clusters. As there is only one surface hypothesis for each image region, there is no indeterminacy in this process.

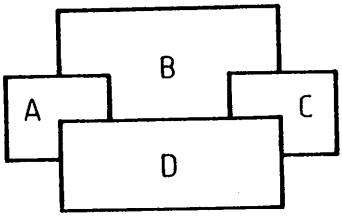


Figure 7–7: Depth Merging Example

Depth Aggregation of Surface Clusters

Another goal of the surface cluster process is to associate all components of an object at some level of surface connectivity. Some aggregation of primitive surface clusters is necessary because self-occlusion may segment the visible portions of an object into several depth levels. In test image 2 (appendix A), the chair seat partially obscures the leftmost chair legs, so the legs and the seat will be in separate surface clusters.

This process is based on the following observation, referring to figure 7–7. If there are four surface clusters: A, B, C, and D, an object might be wholly contained in only one of these, but it might also obscure itself and be in more than one. Hence, reasonable groupings of surface clusters containing whole objects are AB, AD, BC, BD, CD, ABC, ABD, ACD, BCD and ABCD. AC is a less likely grouping because there is no obvious relation between them. Depth aggregated surface clusters are intended to provide the context for complete objects.

Merging all surfaces behind a given surface does not solve the problem. If

only surfaces ABC were present in the above, then this does not produce a containing surface cluster if the true object was ABC. Similarly, merging all surfaces in front fails if both the object and grouping were ACD. Neither of these processes individually produce the correct clusters. To avoid this problem, a more combinatorial solution was adopted. (Whether the production of multiple surface clusters is excessive is addressed in section 7.3.)

Before the algorithm for depth aggregation is given, one refinement is necessary. Rather than consider depth merging for all surface clusters, certain sets of surface clusters can be initially grouped into equivalent depth clusters. These occur when either surface clusters mutually obscure each other, or there is no obvious depth relationship as when across a concave surface boundary. An example of where two surface clusters mutually obscure is with the robot lower arm and trash can surface clusters in test image 1. An example of ambiguous depth relationships is seen in figure 7–8. When these cases occur, all equivalent depth surface clusters can be merged into a single cluster. Then, the combinatorial depth merging process need only consider these equivalent depth surface clusters.

The computation producing the equivalent depth clusters is:

Let:

$\{P_1, ...P_n\}$ be the primitive surface clusters

front(P_i, P_j) is true if P_i is in front of P_j,
which is true if there is a surface in P_i with an
obscuring relation to a surface in P_j

beside(P_i, P_j) is true if P_i is beside P_j,
which is true if not front(P_i, P_j) and not front(P_j, P_i)
and there is a surface in P_i that shares a
concave boundary with a surface in P_j

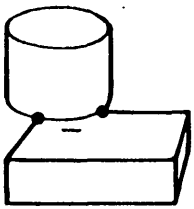


Figure 7–8: Ambiguous Depth Ordering

$\{E_1, ...E_m\}$ be the maximal equivalent depth clusters

$E_i = \{P_{i1}, ...P_{is}\}$

Then:

(1) If $E_i \wedge E_j \neq \phi$ then $E_i = E_j$

(2) for any $P_{ia} \in E_i$, there is a $P_{ib} \in E_i$ such that:

$$\text{front}(P_{ia},P_{ib}) \text{ and } \text{front}(P_{ib},P_{ia})$$

or

$$\text{beside}(P_{ia},P_{ib}) \text{ and not } \text{front}(P_{ia},P_{ib}) \text{ and not } \text{front}(P_{ib},P_{ia})$$

or

$$| E_i | = 1$$

Then, using the same definitions, the depth aggregated surface clusters are sets of equivalent depth surface clusters:

Let:

direct(E_i,E_j)be true if surface cluster E_i is directly
in front of surface cluster E_j,
which occurs if there is primitive surface clusters
$P_{ia} \epsilon E_i$ and $P_{jb} \epsilon E_j$ such that front(P_{ia},P_{jb}).

linked(E_i,E_j) if direct(E_i,E_j) or direct(E_j,E_i)

$\{D_1, ...D_n\}$ be the depth aggregated clusters

$D_i = \{E_{i1}, ...E_{it}\}$

Then:

for any $E_{ia} \epsilon D_i$ there is a $E_{ib} \epsilon D_i$ such that
linked(E_{ia},E_{ib})

The implementation of these constraints is straightforward and leads first to the construction of primitive surface clusters, then to formation of equivalent depth clusters and then to the linking of these into larger depth merged surface clusters. For convenience, the background surface cluster (e.g. the surface that lies behind all others) and the picture frame surface cluster (e.g. the surface that lies in front of all others) are omitted.

These new nodes are linked into the image description graph started in chapter 3 by the following additions:

1. Every surface cluster node is linked to a set of surface hypotheses.
2. Surface clusters are linked into a hierarchy by containment.
3. Surface clusters are linked to chains of boundary elements that separate them from non-surface cluster regions.

7.3 Evaluation: Surface Clusters

This section evaluates the theory of creating surface clusters, as presented in the previous section. The three main points of view are evaluation criteria, performance, and critical discussion.

Evaluation Criteria

The surface cluster formation constraints are simple and logical, except for the laminar surface case and the surface merging based on depth. They are based on obvious three dimensional properties of surface connectivity and object depth ordering. Only minor aspects of the constraints are subject to controversy, because of their generality (discussed in the criticisms section below). In section 7.2, the computations based on the constraints were given. By their simple nature, it is obvious that they meet the constraints. To show that the algorithms, as implemented, execute the desired computation, several examples of program performance will be given. Successful segmentation will demonstrate the merit of the process.

Performance On Test Images

Appendix A shows the two test images used throughout this thesis. Using the surface hypotheses for the two images (see figures 6–11 and 6–12), the surface

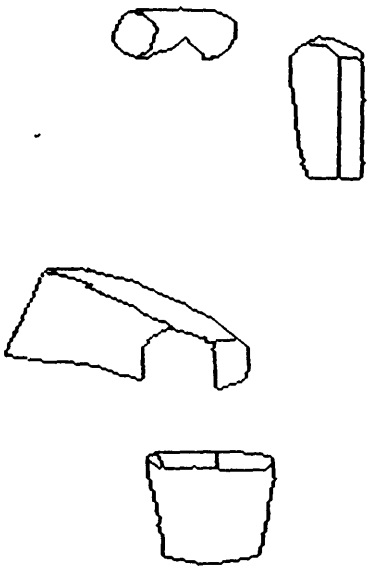


Figure 7–9: Several Primitive Surface Clusters for Test Image 1

clusters for the scenes are shown in figures 7–9 through 7–14. Each image required a minor intervention to produce correct behavior. In the first image, the boundary between the robot body (region 11) and the robot shoulder (region 13) is actually a crack and was forced to appear as a concave boundary. This allowed the body to be depth equivalent with the shoulder, which seems appropriate. In the second image, the two rightmost legs of the chair are so thin that the surface orientation data makes the shape boundary between them and the chair be convex, instead of concave. This was manually corrected, so they could create distinct surface clusters. This problem does not occur with the two leftmost legs because these have an occlusion relationship with the seat.

As can be seen these examples, the surface clusters form object level "chunks" of the image, and correspond to the primitive ASSEMBLYs of the models given in appendix B. Moreover, the pictures above show that ASSEMBLYs are well paired with surface clusters. In table 7–1, there is a listing of the surface cluster to model component correspondences for test image 1. Clearly, the surface cluster

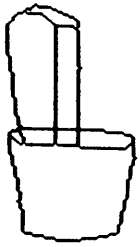


Figure 7–10: Equivalent Depth Surface Cluster for Test Image 1

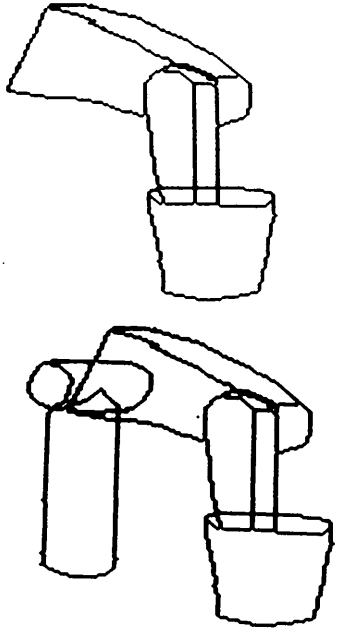


Figure 7–11: Several Depth Merged Surface Clusters for Test Image 1

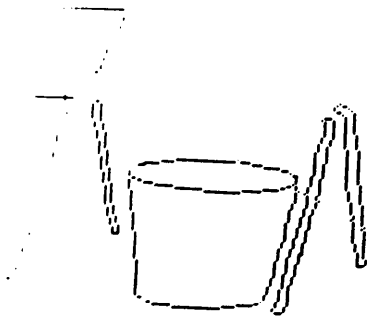


Figure 7–12: Several Primitive Surface Clusters for Test Image 2

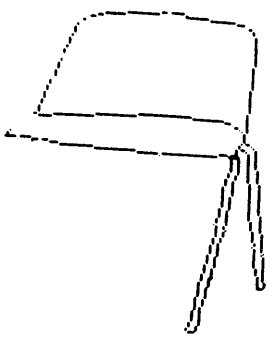


Figure 7–13: Equivalent Depth Surface Cluster for Test Image 2

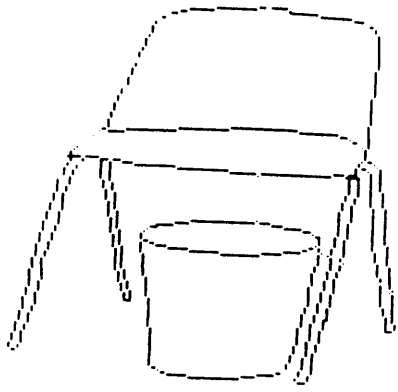


Figure 7–14: Several Depth Merged Surface Clusters for Test Image 2

formation process isolates the key features into what corresponds to structurally based intuitive "objects".

These examples show that the surface cluster formation process is successful in a variety of circumstances and that it is not limited to just planar surface blocks world type objects.

For the example above, the primitive and equivalent depth surface clusters were appropriate. This was also the case in test scene 2. What seems to be a problem is the formation of depth merged surface clusters, which depend on combinatorial groupings of equivalent depth surface clusters. For the two test images, the number of surface clusters in each category were:

IMAGE	PRIMITIVE	EQUIVALENT	DEPTH
1	9	3	6
2	10	1	8

In the first image, the number of depth merged surface clusters was not such a problem as the object also has a strong depth order, so 2 of the 6 corre-

Table 7-1: Surface Cluster to Model Correspondence for Image 1

SURFACE CLUSTER	CLUSTER TYPE	IMAGE REGIONS	MODEL
1	PRIMITIVE	20,21,30	
2	PRIMITIVE	27	
3	PRIMITIVE	16,26	robshldbd
4	PRIMITIVE	8	robbody
5	PRIMITIVE	29	robshldsobj
6	PRIMITIVE	33,34,35,36,37	
7	PRIMITIVE	12,18,31	lowerarm
8	PRIMITIVE	9,28,38	trashcan
9	PRIMITIVE	17,19,22,25,32	upperarm
10	EQUIVALENT	20,21,27,30	
11	EQUIVALENT	8,16,26,29	robshould + robbody
12	EQUIVALENT	9,12,18,28,31,38	lowerarm + trashcan
13	DEPTH	9,12,17,18,19,22, 25,28,31,32,38	upperarm + trashcan
14	DEPTH	8,16,17,19,22, 25,26,29,32	
15	DEPTH	8,9,12,16,17, 18,19,22,25,26, 28,29,31,32,38	link + robot + trashcan
16	DEPTH	8,16,20,21,26, 27,29,30	
17	DEPTH	8,16,17,19,20, 21,22,25,26,27, 29,30,32	
18	DEPTH	8,9,12,16,17, 18,19,20,21,22, 25,26,27,28,29, 30,31,32,38	

sponded to ASSEMBLYs. In image 2, problems are caused because there are three parallel surface clusters behind the "seat back and two legs" equivalent depth surface cluster. Combinatorial grouping causes 8 depth combinations of surface clusters to appear, of which only one is desired (containing the four legs). If more objects had been behind, even more surface clusters (roughly 2^n) would have been created. Hence, the depth aggregated surface cluster formation needs improvement.

Though several surface clusters contained multiple assemblies, this caused no recognition failures, only greater matching effort.

Criticisms

The processes described in this chapter meet their goals when applied to the test images. However, some criticisms can still be raised.

The most obvious criticism concerns performance when applied to rich natural images. Because of the considerably greater quantity of data involved, and possibly missing or erroneous line labels or surface orientation data, there are likely to be failures. The process described in this chapter seems fine for clear laboratory images, but it is hard to predict its performance on natural scenes.

The second criticism questions the validity of the surface segmentation rules. The two problems with these rules are that it is hard to prove that they are sufficiently precise to control over-segmentation, and that they will only apply in appropriate circumstances. Segmentation is a global phenomenon, whereas the rules only apply locally.

A general criticism about the surface cluster is that, as formulated here, it is too literal. A more suggestive process is needed for dealing with natural scenes, where segmentation, adjacency and depth ordering are more ambiguous. The process should be supported by surface evidence, but should be capable of inductive generalization – as is needed to see a complete surface as covering the bough of a tree.

The final criticism concerns the aggregation of primitive and equivalent depth surface clusters to form larger surface clusters. Various alternatives were initially considered. The general goal is to create hypotheses that correspond to the complete visible surface of an object and nothing more. The completion requirement necessitates merging surface clusters. Unfortunately, in the absence of context or object knowledge, there is no information yet to determine whether a surface cluster is related to the surfaces behind. As an object may be associated with only the frontmost two clusters of surfaces, or any two consecutive surfaces, it is likely that the merging process needs to be based on either merging all surface clusters behind the current one, or all possible combinations of consecutive depths. As each surface may be in front of more than one other surface, the latter alternative most likely leads to a combinatorial explosion, whereas the former leads to enlarged surface regions. The combinatorial process, however, probably has better future potential, provided some further merging constraints can be elucidated and implemented. The use of equivalent depth clusters helped control the problem, and as seen in table 7–1, all the primitive and most of the larger surface clusters corresponded with object features.

Extensions

Surface cluster formation could form larger surface regions before surface extension (chapter 6) which would then link the clusters together. This might be useful when the surface is severely fragmented during segmentation (because of real data difficulties), and then partially obscured. The clustering process would merge the connected fragments, and extension would then join them.

In test image 1, the segment between the robot body and shoulder is actually a classical "crack" type line, and the theory should be easily extended to make these segmenting.

Contributions

The research presented in this chapter presented and evaluated rules for producing a new intermediate representation (Surface Clusters) between the $2\frac{1}{2}$D sketch and the model based 3D object hypotheses, which segments the image into blob level, identity independent solids.

Chapter 8

Description of Three-Dimensional Structures

The higher levels of recognition cannot be based on raw image data because of the quantity and lack of appropriate level of description. What reduces the data to manageability is the process of description. Description produces symbolic assertions about the various data entities which are then used by later processes – notably invocation and verification. Invocation uses the descriptions as suggestive evidence, whereas verification uses them as confirming evidence.

Descriptions, as computed here, are simple properties of curves, surfaces and volumes such as curvature, flatness or relative surface angle. They are not reducible to sub-descriptions (i.e. they are not structured) and are computed by special purpose processes.

This chapter presents some descriptions, with the common denominator that they relate to surface and curve structure properties such as surface curvature. Reflectance and illumination based descriptions like surface color, while also useful, will not be considered here. Section 1 discusses the motivations and issues behind the description processes, and section 2 presents the descriptions implemented.

8.1 Motivations

Recognition of complex objects must be based on structures more compact than the raw data. Description produces these by representing masses of data by symbolic [1] assertions, such as "elongation(S,1.2)" or "shape(S,flat)". Two justifications for this are:

1. The data compression makes recognition of arbitrarily placed objects computationally tractable.

2. Recognition processes can be made more independent of the raw data, thus promoting generality.

Because recognition also reduces sets of data to descriptions, what is the distinction between recognition and description? We would be more inclined to say an image curve is "described as straight" than is "recognized as a straight line", whereas the reverse would apply to a person's face. Thus, one criterion is simplicity – descriptions represent simple, consistent phenomena.

Descriptions are specificly reductive – that is, less abstracting. The description "convex" allows approximate local reconstruction of a surface fragment, whereas "face" can hardly be more than generic.

Descriptions are created by low-level processes acting on low-level data, whereas recognition is a higher level symbol matching process. A special purpose process for every conceivable object is not feasible.

If a description is dependent on a conjunction of properties, then it is probably not suitable for use here (e.g. a "square" is a "curve" with equal length "side"s each at "right angle"s). Hence, another criterion is general applicability,

---

[1] Though all description is symbolic at some level, the distinction being made is that the descriptions are based on categorical rather than numerical representations.

because "straight" is a useful description to consider for any boundary, whereas "square" is not.

Description also simplifies relationships. An apple is approximately described as a "reddish" "spheroid". Description allows the bulk of the recognition process to execute uniformly and efficiently, thus leaving detailed comparisons for only a few remaining cases. Here, direct object-image comparisons are made only when estimating spatial positions and verifying object identity.

What properties should be described? Certainly, low level properties such as surface curvature are appropriate. However, at times, we need to deliberately extract object-specific information. A small facial scar might distinguish identical twins. This special purpose analysis should only occur at the highest levels of recognition – when the identity of an object has been reduced to a few directly distinguishable cases for computationally feasibility. Some model-directed "how to look for", "how to discriminate between" and "how to confirm" procedures are probably needed, but this chapter only considers autonomous, low-level, model independent descriptions.

The Specific Descriptions

There are three classes of structures that acquire descriptions: boundaries, surfaces and surface clusters. The surfaces are the surface regions segmented in the input labeled, segmented surface image; segmented boundaries isolate the regions; surface clusters are 3D solids without class identifications. The structures are interrelated, and at times their descriptions depend on their relationships with other entities.

The descriptions are veridical in that they are directly obtained 3D information. Pattern recognition techniques have used 2D projections of 3D properties, but do not fully capture their properties, so discrimination cannot always be correct. To overcome this, people have added constraints available from the real properties of objects, such as the relationship between area and contour ([BRA83],[FIS83]) or from assuming curves are locally planar ([STE83]). With

the additional information available in the surface image, it is possible to obtain 3D information about 3D structures directly. This allows richer and more semantically correct descriptions (i.e. the descriptions are directly related to the true properties of the objects described).

Not all of the properties considered below are viewpoint invariant. This is important because viewpoint invariant properties further the goal of viewpoint independent model invocation and hypothesis completion. In particular, the key invariant properties are local (e.g. curvature) as compared to global (e.g. area). This is only a problem when structures are partially obscured, which affects global properties.

The three structure classes have a variety of descriptions, some of which are listed below. Those with a section number behind the name have been implemented and their computations are described in the given section. Those which are viewpoint invariant are signaled by a "(V)" annotation.

- Boundary descriptions
  - 3D boundary curvature (8.2.1) (V) – estimated using the image path, surface depth and surface orientation.
  - 3D boundary length (8.2.2) – estimated using the image path, surface depth and surface orientation.
  - 3D symmetry axis orientation – may be estimated using the 3D boundary path.
  - parallel boundary orientation (8.2.3) (V) – any 3D parallel segments are found using the boundary orientation.
  - boundary segment orientation (8.2.4) – the angle at which adjacent boundary segments join is estimated using the segment orientations.
- Surface descriptions
  - absolute surface area (8.2.5) – estimated using image areas, surface orientation and distance estimates.

- surface curvature (8.2.6) (V) – the maximum and minimum curvatures for the surface region are estimated from surface orientation and distance measurements. The curvature axis direction is also estimated.

- surface elongation (8.2.7) – the ratio of the maximum to minimum surface region extent is estimated using image dimensions and surface orientation.

• Surface cluster descriptions

- surface angles (8.2.8) (V) – the angle at which two surfaces meet is estimated using local surface orientations.

- relative surface areas (8.2.9) – the proportion a single surface has of the total surface cluster's surface area is estimated using the absolute surface areas (see above).

- surface cluster volume – the volume of the surface cluster may be roughly estimated using the area and depth estimates. The calculation assumes that the depth behind the limits of its visible boundaries is the same as that in front.

- surface cluster elongation – the ratio of the surface cluster's length to cross section width.

- surface cluster elongation axis orientation – the 3D orientation of the surface cluster's long axis.

- surface cluster symmetry – the radial or cross sectional symmetry about the elongation axis may be estimated from the occluding boundaries.

- surface cluster relative volume – the ratio of estimated volumes between adjacent surface clusters.

- surface cluster relative axis orientation – the angle between the elongation axes of two surface clusters.

One piece of meta-information that, while not implemented in this research, might be useful is a data confidence evaluation (such as the mean squared error for a line fitting). Such information could be useful for ranking descriptions or for explaining contradictions encountered during recognition.

The list of 3D descriptions given above is reminiscent of the types of measurements used in traditional 2D pattern recognition approaches to computer vision (e.g. [BAL82],pg254-261). With these, one attempts to measure sufficient object properties to partition the feature space into distinct regions corresponding to single objects. These techniques have been successful for small model sets of simple, distinct unoccluded 2D objects, because the objects can then be partially and uniquely characterized using object-independent descriptions. The research reported in this thesis claims to uniquely identify objects, but only part of the process is aided using such techniques (see chapter 9). The object descriptions will be useful for both "blind" discriminations based only on properties, and on identifications using the descriptions matched to object models.

8.2 The Descriptions

Each subsection below presents one description process. The presentation includes intuitions behind the the description, its computational formulation, examples of its performance and critical discussion.

8.2.1 Boundary Curvature

The calculation of boundary curvature is trivial; the difficulty lies in segmenting each boundary into sections to characterize. Fortunately, the input boundary is labeled both according to the type of segmentation the whole boundary represents (between surfaces) and the type of discontinuities along the boundary (chapter 3). This allows sections to be grouped for description with the goal of finding segments that directly correspond with model boundary sections. Then, 3D data allows direct property computation.

Boundaries are not arbitrary lines, but instead denote segmentations of an object surface. Hence, the surface determines which sections will be described. Boundaries labeled as:

- <back-side-obscuring> are not true object boundaries (relative to the current surface) and thus are not described.

- concave may be true surface boundaries or may occur where another surface rests it; in either case they are described.

- <front-side-obscuring> boundaries may be either orientation discontinuities or tangential generators; in either case they are described.

The circular ordered list of boundary segments surrounding a surface is partitioned into describable groups by the following criteria:

1. If a segment has the label <back-side-obscuring>, then it is deleted.

2. If the point where two segments join is labeled as a boundary segmentation point, then split the list at this point.

3. If two adjacent boundary segments have different labels, then split the list at their junction point.

Assume the figure in figure 8–1 is sitting on a surface, that it is a box with an opening at surface 1, that boundary segment 1 belongs to the background and

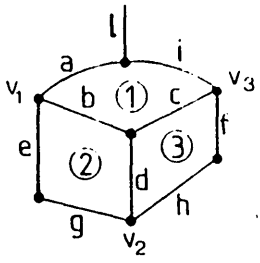


Figure 8-1: Boundary Segment Grouping Example

that the labelings are:

SEGMENT	SURFACE	LABEL
a	1	<front-side-obscuring>
b	1	<back-side-obscuring>
b	2	<front-side-obscuring>
c	1	<back-side-obscuring>
c	3	<front-side-obscuring>
d	2,3	<shape-discontinuity>
e	2	<front-side-obscuring>
f	3	<front-side-obscuring>
g	2	<shape-discontinuity>
h	3	<shape-discontinuity>
i	1	<front-side-obscuring>
l	?	any

VERTEX	IF SEGMENTING JUNCTION
v_1	yes (orientation discontinuity)
v_2	yes (orientation discontinuity)
v_3	yes (orientation discontinuity)
rest	irrelevant

Then, the section for surface 1 is {a,i}. As b and c are <back-side-obscuring>, they are not used. Segments a and i are not separated by any criterion, so are treated as a single section, as is appropriate. For surface 2, each segment is isolated. Between b and d the label changes, as between e and g. Between b and e there is a boundary segmentation point (placed because of an orientation discontinuity in the boundary), as between d and g. Surface 3 is similar to surface 2.

One goal of segmentation was to produce boundary sections with approximately uniform curvature character. Hence, given a boundary section, its curvature can be estimated as follows (refer to figure 8-2):

Let:

$\vec{e_1}$ and $\vec{e_2}$ be the endpoints of the section

$\vec{b}$ be the bisecting point of the section

$\vec{m}$ be the midpoint of the bounding chord

$= (\vec{e_1} + \vec{e_2})/2$

If:

$\vec{m} = \vec{b}$, then the segment is straight,

Otherwise:

$s = |\vec{m} - \vec{b}|$

$t = |\vec{m} - \vec{e_1}|$

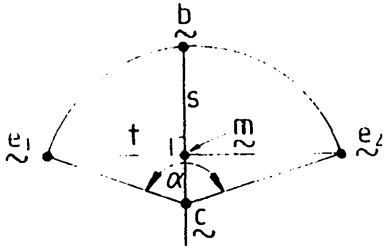


Figure 8 2: Radius Estimation Geometry

And:

$$curvature = 2s/(s^2 + t^2)$$

Several other heuristics were also used to declare nearly straight segments as straight. The curvature estimates for some of the complete boundaries in the test images (using segment labels given in figure 8-3 and 8-4) are given in table 8-1. The "true" values were obtained by hand measurement of the scene objects.

Except for segment 6 in image 1 and 4/5 in image 2, the estimation of curvature is accurate to about 10%. Some straight lines have been classified as slightly curved (e.g. 3/4/5 in image 1) but they received large radius estimates (e.g. 90 cm). The short segments (2 and 17 in image 2) received 0.0 curvature, instead of their high curvature, so short segments should be excluded.

One important point about the curvature is that it is a property not affected by partial occlusion of the surface, provided enough of the surface is visible.

Table 8-1: Boundary Curvature Estimates

IMAGE	REGION	SEGMENTS	ESTIMATED CURVATURE	TRUE CURVATURE
1	26	1	0.131	0.125
1	26	2	0.120	0.125
1	8	3,4,5	0.011	0.0
1	8	6	0.038	0.111
1	8	7,8,9	0.010	0.0
1	9	11,12	0.0	0.0
1	9	13	0.083	0.090
1	9	14,15,16	0.012	0.0
1	9	17,18,19,20	0.054	0.069
2	24	1	0.0	0.0
2	24	2	0.0	2.0
2	24	3	0.0	0.0
2	4	4,5	0.026	0.044
2	4	6	0.0	0.0
2	4	7	0.040	0.044
2	4	8,9	0.0	0.0
2	7	10,11	0.010	0.0
2	7	12	0.093	0.090
2	7	13	0.0	0.0
2	7	14	0.071	0.069
2	23	15,16	0.0	0.0
2	23	17	0.0	2.0
2	23	18,19	0.0	0.0

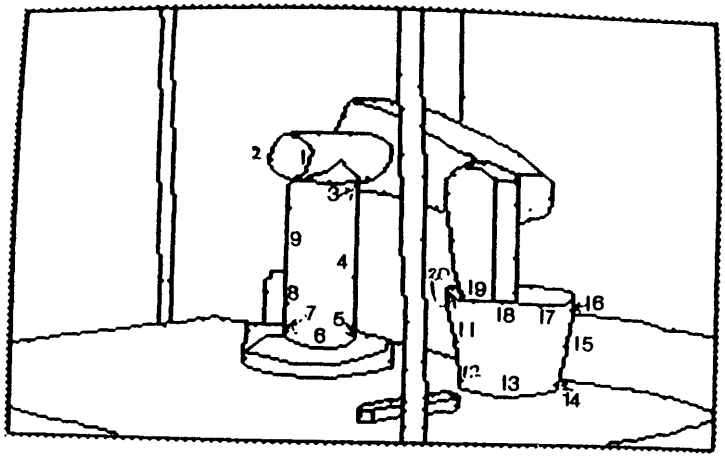


Figure 8–3: Test Image 1 Boundary Numbers

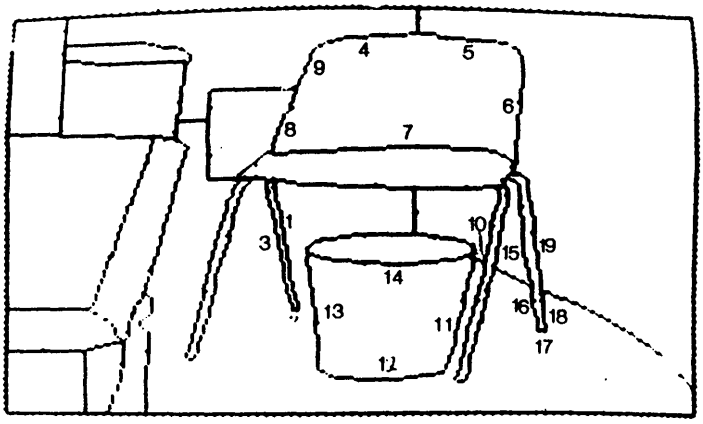


Figure 8–4: Test Image 2 Boundary Numbers

8.2.2 Boundary Length

Given the 3D data, the length of boundary segments can be estimated directly. If the segment is straight, then its length is just the distance between its endpoints. Otherwise, given the groupings described in the previous section, the calculated radius and the geometrical definitions, the length of a curved segment is given as follows:

Let:

$\vec{u}$ be the unit vector in the direction $\vec{m} - \vec{b}$

$\vec{c} = \vec{b} + radius * \vec{u}$ be the center of the arc

$\vec{r_i}$ be the unit vector from $\vec{c}$ to $\vec{e_i}$

Then:

$\alpha = arccos(\vec{r_1} \circ \vec{r_2})$ is the angle subtended by the arc, and

$length = radius * \alpha$

The boundary length estimates for some of the complete boundaries in the test images (using segment labels given in figure 8–3 and 8–4) are:

The average estimation error for boundary length is about 20%, but there are larger errors. The small segments (2 and 17 in image 2) have good absolute accuracy, but deviate because of the poor curvature estimates. The large error for segment 7 in image 2 occurred, in part, because of interpolation errors in constructing a dense surface orientation image. This factor particularly affects boundaries lying on curved surfaces. Surface 24 in image 2 is moderately obscured, which accounts for the shortness of its long segment estimates. The poor estimates for segments 1 and 2 in image 1 result from data errors. On the whole, the estimates are generally acceptable while not accurate. The failures did tend to cause model invocation difficulties because they implied contradictory evidence.

Table 8–2: Boundary Length Estimates

IMAGE	REGION	SEGMENTS	ESTIMATED LENGTH	TRUE LENGTH	% ERROR
1	26	1	31.9	25.2	26
1	26	2	37.0	25.2	47
1	8	3,4,5	51.1	50.0	2
1	8	6	27.3	28.2	3
1	8	7,8,9	46.1	50.0	8
1	9	11,12	28.0	27.2	3
1	9	13	30.1	34.5	12
1	9	14,15,16	25.3	27.2	7
1	9	17,18,19,20	32.7	45.5	28
2	24	1	34.3	45.0	23
2	24	2	1.1	1.5	26
2	24	3	34.2	45.0	24
2	4	4,5	50.3	45.0	12
2	4	6	26.9	29.4	9
2	4	7	42.0	60.6	31
2	4	8,9	26.4	29.4	10
2	7	10,11	27.7	27.2	2
2	7	12	38.3	34.5	11
2	7	13	26.8	27.2	1
2	7	14	42.2	45.5	7
2	23	15,16	41.1	45.0	9
2	23	17	1.1	1.5	26
2	23	18,19	42.8	45.0	5

Table 8-3: Parallel Boundary Group Counts

IMAGE	REGION	BOUNDARIES PARALLEL IN DATA	BOUNDARIES PARALLEL IN MODEL
1	8	2	2
1	9	2	2
1	16	2	2
1	26	1	1
1	29	0	0
2	4	2	2
2	7	2	2
2	9	1	1
2	21	1	1
2	22	1	1
2	23	1	1
2	24	1	1

8.2.3 Parallel Boundaries

A distinctive surface feature is the presence of parallel boundary sections. Hence, one potential description for surfaces is the number of groups of parallel surface boundaries. In this context, parallel means in three dimensions, and requires:

- vectors between endpoints to be parallel, and
- direction of arc curvature to be parallel.

The endpoint vector calculation is trivial given the 3D data. The direction vectors are the $\vec{u}$ defined in the previous section, so the second test is also easy. The results for the test images are given in table 8-3. No errors occurred.

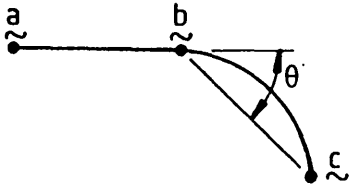


Figure 8–5: Angle Between Boundary Sections

8.2.4 Boundary Join Angles

The angular relationships between sections of a segmented boundary are also a distinctive characteristic of the surface. The angle between the tangents before and after the segmentation point is a potential measurement, but this would not discriminate between a short versus a long arc smoothly joined to straight segment (recalling that the boundary is segmented by orientation and curvature discontinuities). Estimation of the tangent angle is less reliable. Hence, the measurement chosen was the angle between the vectors through the segment endpoints, as illustrated in figure 8–5. Partial occlusion of the boundary will affect this measurement for curved segments. However, if enough of the boundary is visible, the estimate will be close (assuming large radii of curvature).

Some of the join angles for complete object surfaces are reported in table 8–4.

The average angular estimation error is about 0.1 radian, so this process is accurate, considering the error is roughly the same as the data errors.

Table 8-4: Boundary Join Angles

IMAGE	REGION	SEGMENTS	DATA ANGLE	MODEL ANGLE	ERROR
1	26	1 - 2	3.14	3.14	0.0
1	26	2 - 1	3.14	3.14	0.0
1	8	3,4,5 - 6	1.40	1.57	0.17
1	8	6 - 7,8,9	1.79	1.57	0.22
1	9	11,12 - 13	1.73	1.70	0.03
1	9	13 - 14,15,16	1.64	1.70	0.06
1	9	14,15,16 - 17,18,19,20	1.45	1.44	0.01
1	9	17,18,19,20 - 11,12	1.45	1.44	0.01
2	24	1 - 2	1.69	1.57	0.12
2	24	2 - 3	1.42	1.57	0.15
2	4	4,5 - 6	1.61	1.44	0.17
2	4	6 - 7	1.56	1.44	0.12
2	4	7 - 8,9	1.83	1.70	0.13
2	4	8,9 - 4,5	1.27	1.44	0.17
2	7	10,11 - 12	1.42	1.44	0.02
2	7	12 - 13	1.46	1.44	0.02
2	7	13 - 14	1.62	1.70	0.08
2	7	14 - 10,11	1.76	1.70	0.06
2	23	15,16 - 17	1.44	1.57	0.13
2	23	17 - 18,19	1.69	1.57	0.12

8.2.5 Absolute Surface Area

The surfaces of man-made objects typically have fixed sizes, and many natural objects also have surfaces whose sizes fall within narrow ranges. Thus absolute size is a constraint on the identity of a surface. Because we have depth and surface orientation, the absolute surface area of a segmented surface image region can be estimated. Estimation applies to the explicit surface hypotheses which may be larger than the directly related surface image regions because of surface completion (chapter 6). The boundaries of the hypotheses define the corresponding image areas.

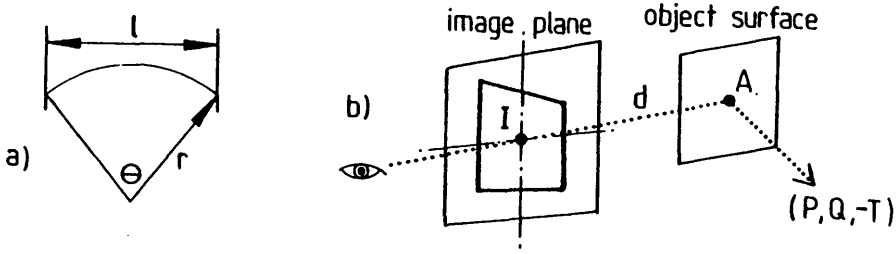
The constraints on the estimation process are:

- The surface region image area is the number of pixels inside the surface region boundaries.
- The image area is convertible to a object slant-projected area at a given depth according to the camera parameters.
- The object slant-projected area is convertible to object surface area using the surface orientation estimates.
- Curved surfaces project to reduced area according to a correction factor. Inversion of this estimates the true surface area.

Approximate curvature correction factors for the two principal curvatures are calculated in the following manner. In figure 8–6 part a, the case of a single dimension is considered. A curved segment of radius R subtends an angle θ . Hence, it has length $R\theta$. This appears in the image as a straight segment of length L. So, the curvature correction factor is:

$$F = R\theta/L$$

where the radius is related to the curvature parameter ($R = 1/C$), the length L is measured from the image (converted from pixels to cm), and the angle θ is



(a) curvature correction (b) projection correction

Figure 8–6: Image Projection Geometries

given by:

$$\theta = 2arcsin(L * C/2)$$

Hence, the complete curvature correction factor is: if $C > 0$

$$F = 2 * arcsin(L * C/2)/(L * C)$$

else

$$F = 1$$

Now, referring to figure 8–6 part b, the absolute surface area computation is derived as follows:

Let:

I be the image area in square pixels,

D be the depth to a nominal point in the surface region,

$(P, Q, -T)$ be the unit surface normal at the nominal point,

G be the conversion factor for the number of pixels per unit length when seen at unit distance,

F, f be the major and minor curvature axis correction factors.

$\vec{v}$ be the unit vector from the nominal point to the viewer

Then, the slant correction is given by:

$$cos(\sigma) = \vec{v} \circ (P, Q, -T)$$

and the absolute surface area is estimated by:

$$A = I * (D/G)^2 * F * f/cos(\sigma)$$

The (D/G) term converts one image dimension from pixels to cm.

In the test images shown in appendix A, unobscured regions from object surfaces with known areas are seen. The absolute surface area for these regions was estimated using the computation described above, and the results are summarized in table 8–5 below. Note that the estimation error percentage is generally small.

From the above discussion, the estimation process is obviously trivial, given the surface image as input. The process is also often accurate, as shown by the results displayed in the above table. The best results occur with the larger surface regions. In part, this is because the effects of spatial quantization on pixel

Table 8–5: Summary of Absolute Surface Area Estimation

TEST IMAGE	IMAGE REGION	PLANAR OR CURVED	ESTIMATED AREA	TRUE AREA	% ERROR
1	8	C	1239	1413	12
1	9	C	1085	1081	0
1	16	C	392	628	38
1	26	P	165	201	17
1	29	C	76	100	24
2	4	C	1416	1390	2
2	7	C	1074	1081	1
2	9	P	1772	1590	11
2	21	P*	79	68	16
2	22	P*	82	68	21
2	23	P*	89	68	31
2	24	P*	49	68	28

* – narrow curved region treated as planar

counts are not as significant. The narrow chair legs in figure 2 had significant error, but their sizes were small.

The use of the nominal point for the whole estimation basis is weak, and a better approach would be to integrate (sum) over all pixels in the region. However, this process is more complex. Since the goal of the process is only to acquire a rough estimate, the implemented approach is adequate. The computation has the largest errors on small or nearly tangential surfaces, because of the reduced image areas. Also, small regions lose a significant number of pixels to the boundary. The depth and orientation estimates were acquired by hand, and are thus a source of error (e.g. region 16). Further, the data values at the nominal point are based on interpolation from nearby data values, and hence have error.

The major conceptual criticism is that natural objects often do not have well defined surface regions, and hence the areas will vary.

8.2.6 Surface Curvature

Because of the surface shape segmentation (chapter 3), each surface region can be assumed to have constant curvature signs and approximately constant magnitude. Using the orientation information, the relative orientation change per image distance is estimated and the distance information is used to convert this to absolute curvature. This description separates surface regions into curvature classes, which provides a first level of characterization. The absolute magnitude of the curvature then provides secondary evidence for the identity of the surface.

Following Stevens ([STE81]) and others, the two principal curvatures, κ_1 and κ_2, are used to characterize the local curvature. These are the maximum and minimum curvatures of the line segments formed by intersecting a normal plane with the surface. (The rotation angles at which these curvatures occur are orthogonal – a property that will be used later.) By the surface shape segmentation assumptions, the shape of each surface region can then be characterized by the two curvatures and their axis orientations. The signs of the two curvatures categorize the surfaces into six possible surface shape groups (table 8-6).

Table 8–6: Surface Shape Classes

	$\kappa_1 < 0$	$\kappa_1 = 0$	$\kappa_1 > 0$
$\kappa_2 < 0$	CONCAVE ELLIPSOID	CONCAVE CYLINDER	SADDLE SURFACE
$\kappa_2 = 0$	CONCAVE CYLINDER	PLANE	CONVEX CYLINDER
$\kappa_2 > 0$	SADDLE SURFACE	CONVEX CYLINDER	CONVEX ELLIPSOID

Turner ([TUR74]) classified surfaces into five different classes (planar, spherical, conical, cylindrical and catenoidal) and made further distinctions on the signs of the curvature. Class membership was determined using local patterns of iso-intensity curves. The work here organizes these classes differently, according to the curvature categories given above. In particular, Turner's cylindrical and conical categories have been merged because they are locally identical. Cernuschi-Frias, Bolle and Cooper ([CER83]) classified surface regionally as planar, cylindrical or spherical, based on fitting a surface shading model for quadric surfaces to the observed image intensities. Both of these techniques use intensity data, whereas directly using the surface orientation data allows local computation of shape. Moreover, using only intensity patterns, the methods give a qualitative evaluation of shape class, instead of absolute curvature estimates.

Brady et al ([BRA84a]) have been investigating a more detailed surface understanding including locating lines of curvature of surfaces and shape discontinuities using 3D surface data. This work gives a more accurate metrical surface description, but is not concerned with the symbolic description of surface segments.

The geometrical aspects of estimating the curvature magnitudes will now be discussed. Initially, the sign of the curvature can be ignored. As figure 8–7 shows, the angle between corresponding surface normals on similar convex and concave surfaces is the same. The two cases can be ultimately distinguished

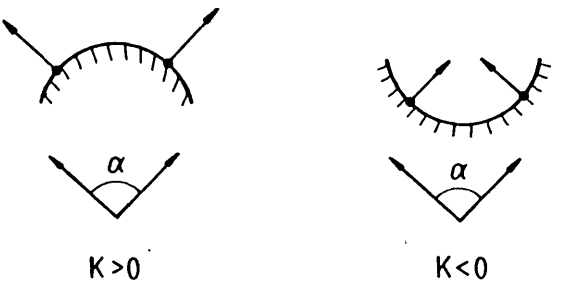


Figure 8–7: Convex And Concave Surface Similarities

because for convex surfaces the surface normal points away from the center of curvature, whereas for the concave case the surface normal points towards it.

Estimating curvature uses the difference in the orientation of surface normals spatially separated on the object surface. The ideal case of a cross-section perpendicular to the curvature axis is shown in figure 8–8. Two unit normals $\vec{n_1}$ and $\vec{n_2}$ are separated by a distance L on the object surface. The angular difference (θ) between the two vectors is given by the dot product:

$$\theta = arccos(\vec{n_1} \circ \vec{n_2})$$

Then, the corrected curvature estimate is:

$$\kappa = 2 * sin(\theta/2)/L$$

This estimates the curvature at a given cross-section orientation.

To find the two principal curvatures, the curvature at all orientations must be estimated. The planar case is trivial, and all curvature estimates are $\kappa = 0$.

The next few paragraphs show that if the curvature is estimated at all orientations using the method above, then the minimum and maximum of these

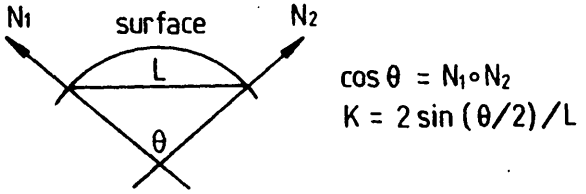


Figure 8–8: Surface to Chord Length Relationship

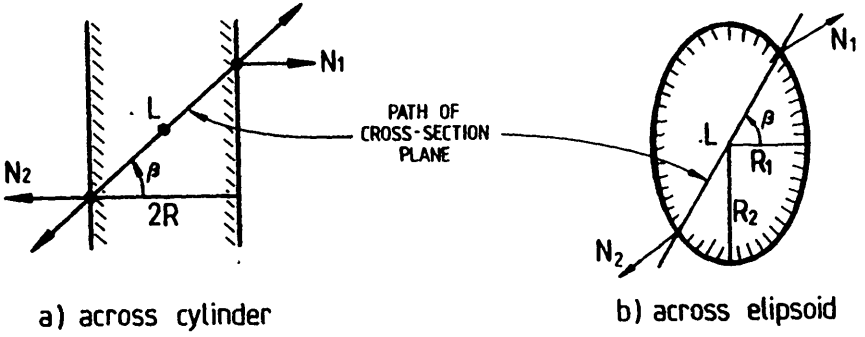


Figure 8–9: Cross-Section Length Relationships

estimates are the principal curvatures. Figure 8–9 part a shows the path of the intersecting plane across a cylinder. For simplicity, assume that the orientation (β) of the plane intersecting the surface starts across the cylinder. Further, assume the cylinder surface is completely observed, so that the points at which the surface normals $\vec{n_1}$ and $\vec{n_2}$ are measured are at the limits of the cross-section. Then, the normals are directly opposed, so θ equals π . The curvature is then estimated using the procedure for a given orientation. While the intersection curve is not always a circle, it is treated as if it were one. (The point is not important, because the estimation is correct for the principal curvatures.)

Let R be the cylinder radius. The chord length L observed at orientation β is:

$$L = 2 * R / \mid cos(\beta) \mid$$

Hence, the curvature estimate (from above) is:

$$\kappa = 2/L = \mid cos(\beta) \mid / R$$

For the ellipsoid case (figure 8–9 part b), the calculation is similar. Letting R_1 and R_2 be the two principal radii (and assuming the third is large relative to these two) the length measured is approximately:

$$L = 2 * R_1 * R_2 / \sqrt{T}$$

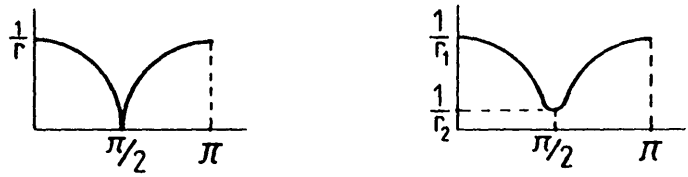
where:

$$T = (R_1 * sin(\beta))^2 + (R_2 * cos(\beta))^2$$

Hence, the curvature estimate (from above) is:

$$\kappa = 2/L = \sqrt{T}/(R_1 * R_2)$$

This analysis gives the estimated curvature versus cross-section orientation β. If $\beta = 0$ were not aligned with a principal curvature axis, then the cross-section would have a shifted phase. In any case, the minimum and maximum values of these estimates are the principal curvatures. The maximum curvature occurs



a) cylindrical surface b) ellipsoid surface

Figure 8–10: Ideal Estimated Curvature Vs Orientation

perpendicular to the major curvature axis (by definition) and the minimum curvature is $\pi/2$ from the maximum. Figure 8-10 shows a graphical presentation of the estimated curvature versus β.

For simplicity, this analysis used the curvature estimated by picking opposed surface normals at the extremes of the intersecting plane's path. Real intersection trajectories will usually not reach the tangential limit of the surface, and instead estimate the curvature with a shorter segment using the method outlined in the beginning of this section. This will produce different curvature estimates for orientations not lying on a major curvature axis. However, the major and minor axis curvature estimates will still be correct, and will still be the maximum and minimum curvatures estimated.

A further point concerns why the separated normal vector approach to curvature estimation was used, rather than using derivatives of local orientation estimates. The justification for this decision is that the larger separation reduces the error in the orientation difference θ when dealing with noisy data. This benefit has to be contrasted with the problem of the curvature changing

over distance. But, as the changes should be small by the segmentation assumption, the estimation should still be reasonably accurate. Another possibility was least-squares fitting of a surface patch, but this was felt to be difficult because three orientation and three curvature parameters would need to be estimated simultaneously. A simpler process would fit a curve to the set of normals obtained along a cross-section at one orientation β.

Given the above geometrical analysis, the implemented surface curvature computation is:

1. Let $\vec{P}$ be a nominal point in the surface image region,

2. Generate the curvature estimate versus β function as outlined above, for cross-sections through $\vec{P}$:

   (a) find cross-section length L

   (b) find surface orientation angle difference θ

   (c) estimate curvature magnitude $\mid \kappa \mid$

3. Fit $\mid cos(\alpha) \mid$ to the curvature versus β function to smooth estimates and determine the phase angle.

4. Extract maximum and minimum curvature magnitudes

5. At maximum and minimum curvature orientations, check direction of surface normal relative to surface (see figure 8–7).

   (a) if into surface, then $\kappa < 0$

   (b) if out of surface, then $\kappa > 0$

The estimation of the major axis orientation for the surface regions is now easy. (The major axis is that about which the greatest curvature occurs.) The second axis of curvature is constrained to be perpendicular to the major axis and the surface normal at $\vec{P}$. The plane case can be ignored, as it has no curvature.

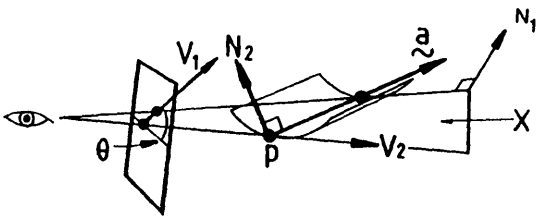


Figure 8–11: Curvature Axis Orientation Estimation (Find Axis Plane)

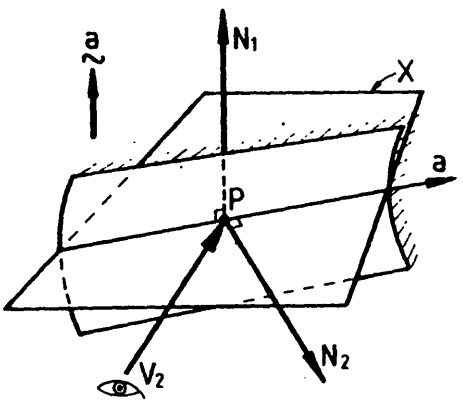


Figure 8–12: Curvature Axis Orientation Estimation (Find Vector)

Figure 8–11 and 8–12 illustrates the geometry for the axis orientation estimation process.

The plane through the major axis $\vec{a}$ and the viewpoint intersects the surface in a line lying along the major axis (see figure 8–11). This plane can be characterized by the normal $\vec{n_1}$, which is calculated as follows. Let θ be the orientation in image plane of major axis. Then the vector $\vec{v_1}$ lies in this plane:

$$\vec{v_1} = (cos(\theta), sin(\theta), 0)$$

Further, the vector $\vec{v_2}$ from the viewer to the nominal point $\vec{P}$ also lies in this plane. This vector can be constructed by inverting the projection relationship. As both of these vectors lie in the desired plane, and the two must be distinct ($\vec{v_1}$ is seen as a line, whereas $\vec{v_2}$ is seen as a point), the normal to the plane is:

$$\vec{n_1} = \vec{v_1} \times \vec{v_2}$$

The major curvature axis lies in this plane. It also lies in a plane parallel to the plane tangential to the surface at the nominal point $\vec{P}$ (see figure 8–12) for constant radius cylinders and ellipsoids. If the surface orientation vector at this point is the vector $\vec{n_2}$, then the major axis $\vec{a}$ is:

$$\vec{a} = \vec{n_1} \times \vec{n_2}$$

as the axis $\vec{a}$ must be perpendicular to both of the planes' normals.

The curvature and axis orientation estimation process was applied to the test images shown in appendix A. The curvatures of all planar surfaces were estimated correctly as being zero. The major curved surfaces are listed in tables 8–7 and 8–8 below, with the results of their curvature and axis estimates. (If the major curvature is zero in table 8–7, then the minor curvature is not shown.) In table 8–8, the error angle is the angle between the measured and estimated axis vectors.

The estimation of the surface curvature and axis directions is both simple and accurate, as evidenced by the above discussion and the results. Again, as the depth and orientation estimates were acquired by hand, this is one source of

Table 8–7: Summary of Surface Curvature Estimates

TEST IMAGE	IMAGE REGION	MAJOR(MJ) MINOR(MN)	ESTIMATED CURVATURE	TRUE CURVATURE
1	8	MJ	.127	.111
		MN	0	0
1	9	MJ	.081	.078
		MN	0	0
1	12	MJ	0	0
1	16	MJ	.136	.125
		MN	0	0
1	18	MJ	0	0
1	25	MJ	.139	.130
		MN	0	0
1	26	MJ	0	0
1	29	MJ	.124	.125
		MN	0	0
1	31	MJ	.089	.090
		MN	0	0
2	4	MJ	-.037	-.044
		MN	0	0
2	7	MJ	.082	.0787
		MN	.001	0
2	9	MJ	0	0
2	16	MJ	-.070	-.0787
		MN	.003	0
2	21	MJ	0	0
2	22	MJ	0	0
2	23	MJ	0	0
2	24	MJ	0	0

Table 8–8: Summary of Curved Surface Curvature Axis Estimates

TEST IMAGE	IMAGE REGION	ESTIMATED AXIS	TRUE AXIS	ERROR ANGLE
1	8	(0.0,0.999,0.0)	(0.0,1.0,0.0)	0.02
1	16	(-0.99,0.10,0.02)	(-0.99,0.0,0.1)	0.17
1	25	(-0.98,0.11,0.14)	(-0.99,0.0,0.1)	0.16
1	31	(-0.99,-.03,0.11)	(-0.99,0.0,0.1)	0.10
1	9	(-0.09,0.99,-0.07)	(0.0,1.0,0.0)	0.12
1	29	(-.04,0.99,0.0)	(0.0,1.0,0.0)	0.05
2	4	(0.08,0.99,-0.03)	(0.0,1.0,0.0)	0.09
2	7	(0.05,0.99,-0.07)	(0.0,1.0,0.0)	0.09
2	16	(-0.02,0.99,0.07)	(0.0,1.0,0.0)	0.08

error in the results. Another source is the inaccuracies caused by interpolating depth and orientation estimates between measured values.

The major weak point in this analysis is that the curvature can vary over a curved surface segment, whereas only a single estimate is made (though the segmentation assumption limits its variation). Choosing the nominal point to lie roughly in the middle of the surface will help average the curvatures, and it will also help reduce noise errors by giving larger cross-sections over which to calculate the curvature estimates. A minor extension is needed for estimating the 3D axis of cones (as compared to cylinders).

8.2.7 Surface Elongation

The elongation of surface regions is also a distinguishing characteristic. This has been a traditional pattern recognition measurement applied to 2D objects, but now the descriptions can also be obtained for 3D objects, as the true 3D dimensions can be used instead of the projected dimensions.

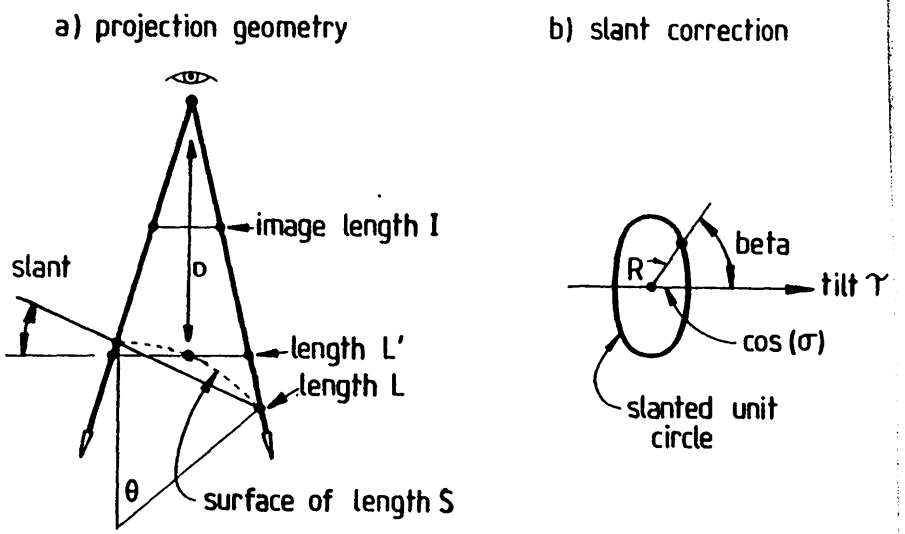


Figure 8–13: Cross-Section Length Distortions

The four factors involved in the estimation of the surface's dimensions are: the image region's dimensions, the surface slant relative to the viewer, the curvature of the surface, and the distance from the viewer. The elongation value is the ratio of the longest to the shortest dimension of the surface.

The true cross-section path length should be calculated for all paths across the surface, but this is unnecessarily detailed. Instead, an estimate of the cross-section width is computed about a central point, and it is adequate to roughly characterize the elongation of the surface.

Figure 8–13 part a shows a sketch of the viewing relationship at one cross-section through the surface. By the discussion in section 8.2.5, the surface length S is approximately related to the chord length L as:

$$S = L * \theta / (2 * sin(\theta/2))$$

Then, if the cross-section is slanted away from the viewer by an angle α , the observed slanted length L' is approximately related to the chord length L (assuming the viewing distance is large) by:

$$L' = L * cos(\alpha)$$

Finally, the observed image length I for the surface at depth D is proportional to the slanted length L' by:

$$I \propto L'/D$$

This analysis needs to be modified to account for the effect of slant compression at angles other than the tilt angle. Figure 8–13 part b shows the geometry used for the following analysis. This figure shows a unit circle compressed by a slant angle σ in the direction τ and orthographically projected onto the image plane. Elementary trigonometry and algebra show that the observed length V at the angle β follows:

Let:

β = the angle relative to the tilt axis τ

σ = slant angle

Then:

$$V = 1/\sqrt{1 + (tan(\sigma) * cos(\beta))^2}$$

The computation of the elongation value is then:

Let:

$\vec{N}$ be a nominal point in the center of the surface image region

$w(\alpha)$ be the image cross-section width at image angle α about $\vec{N}$

$\theta(\alpha)$ be the change in surface orientation across the cross section at image angle α

$(P, Q, -T)$ be the unit surface normal at $\vec{N}$

D be the distance from the viewer to $\vec{N}$

G be the conversion factor for the number of image pixels per unit length when seen at unit distance

$\vec{v}$ be the unit vector pointing to the viewer from the point on the object surface corresponding to $\vec{N}$

Then, the tilt angle is:

$$\tau = arctan(Q/P)$$

the relative slant direction β is:

$$\beta = \alpha - \tau$$

the slant angle σ is:

$$\sigma = arccos(\vec{v} \circ (P, Q, -T))$$

the slant correction factor is:

$$M = \sqrt{1 + (tan(\sigma) * cos(\beta))^2}$$

the projected chord length L' is:

$$L'(\alpha) = w(\alpha) * (D/G)$$

the unprojected chord length L is:

$$L(\alpha) = L'(\alpha) * M$$

and the estimated 3D cross-section is:

$$cross(\alpha) = L(\alpha) * \theta(\alpha)/(2 * sin(\theta(\alpha)/2))$$

Table 8-9: Summary of Estimated Elongations

TEST IMAGE	IMAGE REGION	PLANAR OR CURVED	ESTIMATED ELONGATION	TRUE ELONGATION
1	8	C	3.3	2.0
1	9	C	1.8	1.7
1	16	C	2.9	1.5
1	26	P	1.4	1.0
1	29	C	3.6	3.1
2	4	C	2.2	2.1
2	7	C	1.8	1.7
2	9	P	1.1	1.0
2	21	P*	24.2	28.5
2	22	P*	46.5	28.5
2	23	P*	5.3	28.5
2	24	P*	19.5	28.5

* – narrow curved region treated as planar

Finally, the elongation is:

$$E = max(cross(\alpha))/min(cross(\alpha))$$

The elongations for all unobscured image regions that correspond to model segments are listed in table 8-9.

These results show that the estimation process gives approximate results when applied to unobscured regions. In part, small regions should be more affected because single pixel errors are significant, but this is not always the case. The chair leg regions almost always received large elongations, but the actual values were inaccurate.

The major theoretical weakness is that the concept of elongation is best applied to planer surfaces. Another weakness of the process is that it only

estimates the dimensions based on reconstructions about a single point, which produces lower bounds for the maximum and upper bounds for the minimum cross-section. Prematurely encountering the boundary during the cross-section evaluation accounts for the abnormally low estimate for image 2 region 23. This results in an estimate that will be lower than the true elongation. The dimensions of a bounding rectangle rather than the lengths of cross-section paths would probably improve the values. Further, because the curvature correction process assumes uniform curvature along the cross-section path, the reconstructed length L will vary from the true length. Another source of error is from the hand measured surface data and the surface interpolation. However, these and the other approximations were felt to be practically justifiable and that the above results show evidence can be acquired from this data. Perhaps the best use of this type of evidence would be to roughly quantise the valuations into: compact, squashed, elongated and stick, with some overlap on the ranges producing each description.

8.2.8 Surface Angles

Three dimensional information facilitates new description types, such as the angle between surface regions. Given the surface orientations available from the surface image, and some elementary geometry, it is possible to directly compute the angle at which two surfaces meet (the angle that the solid portion of the junction subsumes).

This description is useful for two reasons: (1) extra information is always useful for recognition and (2) the measurement is dependent on the surface's relationship with its neighbors, whereas many other descriptions relate only to the structure in isolation. Hence, context begins to have an effect on identification. A generic surface shape may not have any constraints on its relationships with neighboring surfaces, in which case these descriptions would be ignored. However, an instance of the surface in an object context would have these constraints, and so the extra information would lead to higher likelihoods of model

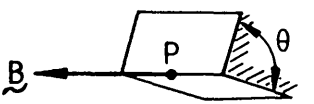


Figure 8–14: Two Adjacent Surfaces

invocation (chapter 9). A further point about this description is that it is only applied to surfaces adjacent across a shape boundary and so emphasizes group identification. (Surfaces across an occluding type boundary are not directly related.)

The factors involved in the description's calculation are the orientation of the surfaces, the shared boundary between the surfaces and the direction to the viewer. As the boundary is visible, the viewer must be in the unoccupied space sector between the two surfaces.

Because surfaces can be curved, the angle between them may not be constant along the boundary. It is assumed that this angle will not vary significantly without the introduction of other shape segmentations, and so the calculation obtained at a nominal point is taken to be representative.

Figure 8–14 shows two surfaces meeting at a boundary. Somewhere along this boundary a nominal point $\vec{P}$ is chosen and also shown is the vector of the boundary direction at that point ($\vec{B}$). Through this point a cross-section plane

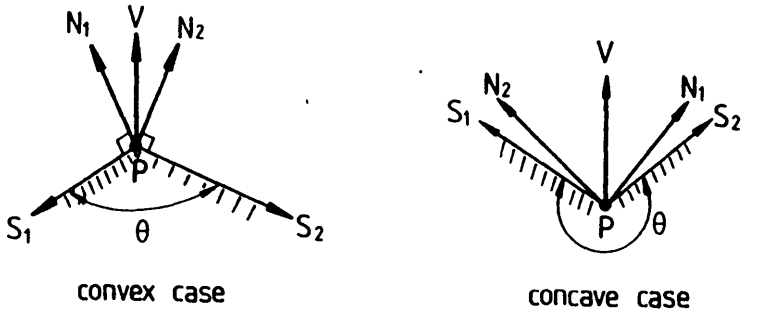


Figure 8–15: Surface Normals and the Two Surface Cases

is placed, such that the normals ($\vec{n}$) for the two surfaces lie in the plane. Figure 8–15 shows the two cases for this cross-section.

The essential information that determines the surface angle is the angle at which the two normals meet. However, it must also be determined whether the surface junction is convex or concave, which is the difficult portion of the computation. The details of the solution are seen in figure 8–15 above. Vectors $\vec{S}_1$ and $\vec{S}_2$ lie along the respective surfaces. By definition, the vector $\vec{B}$ is normal to the plane in which the $\vec{n}$ and $\vec{S}$ vectors lie. Hence, each individual $\vec{S}$ vector is normal to both the corresponding $\vec{n}$ vector and the $\vec{B}$ vector, and can be calculated by a cross product.

The angle at which the $\vec{S}$ vectors meet is also the surface junction angle, but the convex/concave distinction still needs to be made. Because the boundary must be visible, the angle between the view vector $\vec{v}$ from the nominal point to the viewer and each of the surface vectors $\vec{S}$ must be less than π. Hence, the magnitude of the angle between the two vectors is guaranteed to represent open

space. As a result, the solid space is 2π minus these two open spaces. This computation is summarised below:

Let:

$\vec{P}$ be a nominal point on the boundary between the two surfaces

$\vec{n_1}$, $\vec{n_2}$ be the two surface normal vectors at $\vec{P}$

$\vec{v}$ be the vector from the nominal point P to the viewer

Then, the boundary vector $\vec{B}$ is:

$$\vec{B} = \vec{n_1} \times \vec{n_2}$$

and the surface vectors $\vec{S_i}$ are:

$$\vec{S_i} = \vec{B} \times \vec{n_i}$$

These $\vec{S}$ vectors may face the wrong direction, e.g. away from the surface. To get the direction correct, a track is made from the point $\vec{P}$ in the direction of both $\vec{S}$ and $-\vec{S}$ projected. One of these should immediately enter the surface region, and this is assumed to be the correct $\vec{S}$ vector. Given this, the surface angle is:

$$\theta = 2\pi - \mid arccos(\vec{v} \circ \vec{S_1}) \mid - \mid arccos(\vec{v} \circ \vec{S_2}) \mid$$

The true and estimated surface angles for the modeled objects are summarised in the following table. Further, only rigid angles between surfaces in the same primitive surface clusters are reported (these being the only evidence used).

The estimation procedure is accurate for orientation discontinuities. The major source of errors for this process is a result of measuring the surface orientation

Table 8–10: Summary of Estimated Surface Angles

TEST IMAGE	IMAGE REGIONS	ESTIMATED ANGLE	TRUE ANGLE	ERROR	NOTE
1	16,26	1.47	1.57	0.10	
1	16,29	2.96	3.14	0.18	
1	12,18	1.53	1.57	0.04	
1	12,31	1.60	1.57	0.03	
1	18,31	2.03	2.14	0.11	
1	17,25	2.09	3.14	1.05	*
1	17,22	1.56	1.57	0.01	
2	4,9	4.69	4.71	0.02	

* – large error across a curvature discontinuity

vectors by hand, and interpolating the value to the nominal point. This would have contributed to the error at the curvature discontinuity, where interpolation probably flattened out the surface.

8.2.9 Relative Surface Area

While the absolute size constrains isolated surface identities, relative surface area constrains it in an object context. Because a surface generally appears with a known set of other surfaces, the *a priori* range of the relative proportion of the total visible surface area can be determined. The precise relative area is, in theory, determinable for all viewing positions, but in practice only the proportion limits defined by the representative positions need be considered. The advantage of using the absolute area in this calculation is that, if the topology of the viewed surface does not change, the relative surface areas do not change either.

The relative surface area calculation is trivial, once the individual component's absolute areas have been calculated. The primitive surface cluster is a

Table 8–11: Summary of Relative Surface Area Estimation

TEST IMAGE	IMAGE REGION	PLANAR OR CURVED	IMAGE CONTEXT	ESTIMATED PROPORTION	VALID RANGE
1	8	C	8	1.00	1.00
1	9	C	9,28,38	0.92	0.6 - 1.0
1	16	C	16,26	0.70	0.76
1	26	P	16,26	0.29	0.24
1	29	C	29	1.0	1.0
2	7	C	7,16	0.80	0.6 - 1.0

blob-level representation for an object (chapter 7), and the surface clusters are used as the relative surface area data contexts.

Table 8–11 summarizes the results of the relative surface area calculation for the same image regions as in table 8–5. Again, the same good performance is noted as in the previous section. A point to note about the relative area is that valid evidence can still be computed even if only the relative distance (as compared to the absolute distance) to the object's surfaces is available. This point also holds for objects with fixed geometry, but variable size: the relative proportion of the total size remains the same. This evidence does not occur as often, because the relative values are calculated only within a primitive surface cluster. By its definition, only the trash can qualify for this in image 2, so little distinguishing evidence of this type was obtained.

Final Comments

This chapter:

- showed that by using surface image data a variety of general identity-independent three dimensional properties were directly computable,
- showed the properties applied to curves, surfaces and surface clusters,

- gave the computation for some of them, and
- showed that the properties could usually be calculated accurately.

Chapter 9

Model Invocation

One important and difficult task for a general model based vision system is invoking the correct model. Because of the potentially huge number of possible objects, it is imperative that only a few serious candidates are selected for detailed consideration. Even a modest industrial vision system may have 100 distinct objects in its repertoire, so the problem is not limited just to the research domain. Visual understanding must include a non-attentive element, because all models need be considered, yet active, direct comparison is computationally infeasible. Further, data errors, generic models and previously unseen objects require selecting models that are "close" to the data, so invocation must also address this problem.

This chapter presents a solution that embodies ideas on association networks, object description and representation, and parallel implementations to solve the problem. In the first section, the relevant aspects of the problem are discussed. The second presents a theoretical formulation of the proposed solution. The third discusses some interesting algorithmic points, and the last evaluates the theory.

9.1 Motivations: Considerations on the Invocation Process

The primary motivation for the invocation process is that model-based vision is computationally intractable without reducing the large set of objects potentially explaining a set of data to a small subset of serious candidates. Since every object is a potential candidate, and there may be 10000 – 100000 distinct objects in a competent general vision system's range, the problem is too large for model-directed comparisons of every known object in every viewpoint to the data, even when the potentially massive parallelism of the brain or of VLSI are considered. Yet, every object must also be accessible as a candidate because an image structure could have any interpretation. So, the solution must consider both efficiency and completeness of access.

There is also a more crucial competence aspect to the problem. We are capable of (loosely) identifying previously unseen objects, based on similarity to known objects. The problem also occurs with flexible objects seen in new configurations, or with incompletely visible objects (e.g. occlusion) or object variants (e.g. flaws, generics, new exemplars). Hence, nearby "similar" models should be invoked to help start identification, where "similar" means sharing some features or having identically arranged substructure.

One difficulty with invocation is deciding which aspects of the problem are related to general cognition, to general vision, to surface-based vision or to specifically learned objects.

At one extreme, there appear to be invocation cues that are object-specific, such as the particular shade of blue light used by police and emergency vehicles. It almost never appears in any other context, so can become a distinct symbol. Further, because of its uniqueness, one can immediately invoke the "emergency/police" context without additional evidence. This is a problem because vision research tries to focus on the general aspects of vision, where features

and functions are common, and only the configurations are distinct. This cue is object-specific and is likely to be learned as compared to being a visual primitive. Specific invocation cues may themselves be reducible to lower level descriptions that are more amenable to general visual processing.

At the other extreme, some inputs to invocation have to be strictly visual and built-in (i.e. not learned). Most notably, there are the primitive descriptions autonomously generated by low level visual processing (e.g. those given in the last chapter). Not all cues need be visual (e.g. the words "devilish fruit" for an apple), but these are not considered here.

This thesis is only concerned with vision, but the invocation results have more general applicability. Any form of symbolic description requires accessing the correct symbol. So, the model invocation problem is also a general cognitive problem, with the following aspects:

- low level symbolic assertions are produced for the current input whether from an external (e.g. raw sense data) or internal (e.g. self-monitoring) source,

- higher level concepts/symbols tend to be semi-distinctly characterizable based on "configurations" of lower level symbolic descriptions,

- there are many potential higher level symbols, but only a small subset should be selected for closer consideration when matching a symbol to a set of data,

- the importance of a particular concept in invoking another is dependent on many factors, including structure, generics, experience and context, and

- symbols "recognised" at one description level (either primitive or through matching) become usable for invoking more complex symbols.

Examples of this in a non-vision context might be something like an invocation of a Schankian fast-food restaurant schema ([SCH75]), a selection of a

likely person as the source of an unidentified phone call, or recognising words in speech.

The theory proposed in section 9.2 incorporates the following characteristics of invocation:

Invocation Is Suggestion

Invocation is the undirected convergence of clues that suggest identities for explaining data. On the street, one occasionally sees a person with a familiar figure, face and style, but who on closer inspection turns out not to be the acquaintance. The clues suggest the friend, but direct examination contradicts.

Invocation also supports the "seeing" of nonexistent objects, as in Magritte's surrealist paintings, where configurations of features give the impression of one object while being another. Figure/ground reversals and ambiguous interpretations such as the vase and faces illusion could occur when multiple invocations are possible, but only a single interpretation is held at any instant, because of mutual inhibition, as suggested by Arbib ([ARB79]).

Invocation Is Via Association

Invocation is computed through association. The association principle is based on suggestion, rather than confirmation. For example, a red spheroid might suggest an apple, even though it is a cherry. A better example is seen in the Picasso-like picture drawn in figure 9–1. Though many structural relationships are violated, there are sufficient suggestions of shapes, correct subcomponents and rough relationships for invoking a human face and figure model.

While there are many types of association between visual concepts, two key ones are mediated by generic and component relationships as discussed below. Other association types include context (robots are more often found in factories than in trees) or temporal (a loud bang is often heard after the flash of lightning).

Figure 9–1: Picasso-Like Figure Invokes Human Model

Associations Have Varying Importances

The importance of a particular feature in invoking a model is relative to the feature, model, context and viewing system.

Some objects share common features, such as planar faces in blocks world scenes. Other objects have distinct features, such as the shape of the Eiffel Tower, an ear, or the illumination from the sun. Hence, some features may strongly constrain the set of potential models, whereas others may only do so weakly.

Context is also important, because the a priori likelihood of discovering an object influences the importance of a feature. Wheels (generic) seen in a garage are more likely cues for automobiles than when seen in a bicycle shop. In figure 9–2 part a, there is a standard pyramid that is unlikely to invoke any models other than its literal interpretation. Yet, in figure part b, the same pyramid invokes the "nose" model acceptably. Obviously, the immediate, higher level, context influences the likely models invoked for a structure. The viewing system

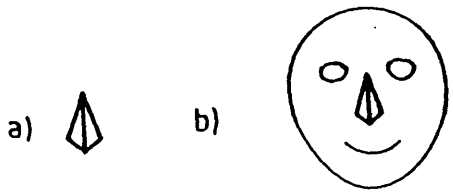


Figure 9–2: Pyramid in Face Context Invokes Nose Model

is also a factor, because its perceptual goals influence the priorities of detecting an object. (Industrial inspection systems concentrate on a few known, distinctive features of a few objects, rather than the objects as a whole.)

Statistical analysis could determine the likelihood of a feature in a given context, but would be useless in determining the importance of the feature in invoking the model. Further, because contexts may change, it is difficult to assess the object's a priori distribution. Finally, experience over time can be expected to change feature importance as factors become more or less significant. Hence, importance assessment seems more properly a learning than an analysis problem.

Evidence Comes from Observed Properties

Direct evidence is based on symbolic description of features and properties of structures. This is information extracted from visual input, as compared to indirect evidence from circumstantial associations, or from generic or structural relationships. Because of the requirements of invocation, the evidence should

be generic, though it may include some simple parameters. An example of this is surface shape evidence, which may have zero, one or two axes of curvature (parameterized by the degree of curvature). The relevant aspects of evidence are type and value (which is also appropriate for verification).

The properties used here are absolute measurements, with their acceptable ranges constrained by bounds (like ACRONYM [BRO81]). A better approach follows Marr ([MAR82]), who suggested properties should be recorded using something like quantized value range descriptors that decompose to finer descriptions as required (e.g. 30% of the major axis length units refining to 20% lengths). The use of units then allows value testing through symbol matching rather than through inequality testing. The concept of tuned "descriptor" units producing the output descriptions is attractive, but scale is not yet well understood, nor is there much evidence for the range of parameters that should be included in a unit. Angular quantities are intrinsically size invariant, but feature dimensions cause problems. A relative size descriptor requires description in a global rather than local context, which reduces modularity and also throws away the absolute size information. Absolute size descriptors make generic identifications difficult: Michaelangelo's David, a normal human and a "GI-Joe" doll all have the male-human identity even though their structural scale varies by about 20.

Evidence is acquired in "representative position" ([COW83]). Features that remain invariant during the 3D to 2D projection process, with the observer in general position, are few (e.g. color). As this thesis is concerned with surface shape properties, the potential evidence is more limited and includes angles between structures, axis orientations, relative feature sizes and relative feature orientations when unobscured. Invocation features should be usually visible. When they are not, invocation may fail – as in Marr's picture of a bucket from above ([MAR82], pg. 316). There may be alternative invocation features for the privileged viewing positions.

Evidence Comes from Generic Relationships

An object typically has several generalisations and specialisations, and evidence for its invocation may come from any of these other hypotheses. For example, a generic chair could generalize to "furniture", or "seating structure", or specialise to "dining chair" or "office typing chair". Because of the unpredictabilities of evidence, it is conceivable that any of the more generalized or specialized concepts may be invoked before the generic chair. For the more general, this may occur when occlusion leaves only the facts of seating surface and back support prominent. Further, these are the key aspects of the concept, so the simpler structure might achieve a higher plausibility given the evidence. Conversely, observation of a particular distinguishing feature may lead to invocation of the more specific model first.

In either case, evidence for the categorically related structures gives support for the structure.

This has some superficial relationships with the generic identification scheme in ACRONYM ([BRO81]). There, identification consists of maximal descent through a specialization hierarchy with the object meeting all constraints at each level. The hierarchy is similar, and the notion of refining constraints is also similar. The suggestion in this subsection differs in that: (1) the goal is suggestion, not verification, so the property constraints are not strict, and (2) the flow of control is not general to specific: identification could locally proceed in either direction in the hierarchy.

Evidence Comes from Component Relationships

The presence of an object's subcomponents suggests the presence of the object. If we see the legs, seat and back of a chair, the whole chair is also likely but not guaranteed, as we could be seeing an unassembled set of chair parts. Hence, verified or highly plausible subcomponents influence the plausibility of the object. The reverse should also occur. If we are reasonably sure of the chair, such as from having found several subcomponents of the chair, then this should enhance

Figure 9–3: Identified Subcomponents Invoke Models

the plausibility that nearby leg-like structures are chair legs. This is useful when such structures are partially obscured, and so their direct evidence is not as strong.

Figure 9–3 shows an abstract head. While the overall face is unrepresentative, the head model is invoked because of the grouping of correct subcomponents.

Evidence Comes from Configurations of Components

Configurations of subcomponents have two aspects: (1) only a subset of subcomponents is visible from a particular viewpoint, and (2) the spatial (or temporal) distribution of subcomponents can suggest models as well. The first case implicates having key feature groupings when integrating evidence. For a chair, one often sees the top of the seat and the front of the back support, or the bottom of the seat and back of the back support, or the top of the seat and the back of the back support, but seldom any of the other twelve groupings of the four subcomponents. These groupings are directly related to the major visibility re-

gions in the visual potential scheme suggested by Koenderink and van Doorn ([KOE77]). They showed how the sphere of all potential viewing positions of an object could be partitioned according to what components were visible in each partition and when various features became self-obscured. Minsky ([MIN75]) suggested distinguishable viewer-centered feature groupings should be organized into separate frames for recognition. The scheme proposed here is less literal, in that precise features are not expected, minor features and the exact representation of self-occlusion are ignored (which would create many more partitions) and the relationships between different viewpoints is not made explicit.

Figure 9–4 shows the sphere of viewpoints partitioned into the topologically distinct regions for a trash can. At a particular scale, there are three major regions: outside bottom plus adjacent outside surface, outside surface plus inside surface and outside, inside and inner bottom surfaces. There are other regions that could also be included as an "other viewpoint" category, but these are the ones of significance.

With more articulated or intricate objects, all regions in a literal view potential scheme are likely to be small, so some decisions are needed about what features to include, and how to choose reasonable visibility groups. In the results described in this thesis, they were chosen by hand.

During invocation, these groupings will be used to collect evidence, and the success of a grouping will imply a rough object orientation. Hinton ([HIN81]) proposed tuned orientation estimation units that performed this task in 2D. Here, the grouping is the important result, which will be used to make initial structure assignments in hypothesis completion (chapter 10), from which orientation is estimated directly. Of course, high plausibility of a particular grouping makes likely a particular object orientation.

The second aspect of configurations is how the placement of components, rather than their identity, suggests the model. Figure 9–5 shows the converse of figure 9–3, where all subcomponents have the wrong identity but, by virtue of their position, suggest the face model. This aspect of invocation was not explored further in this thesis.

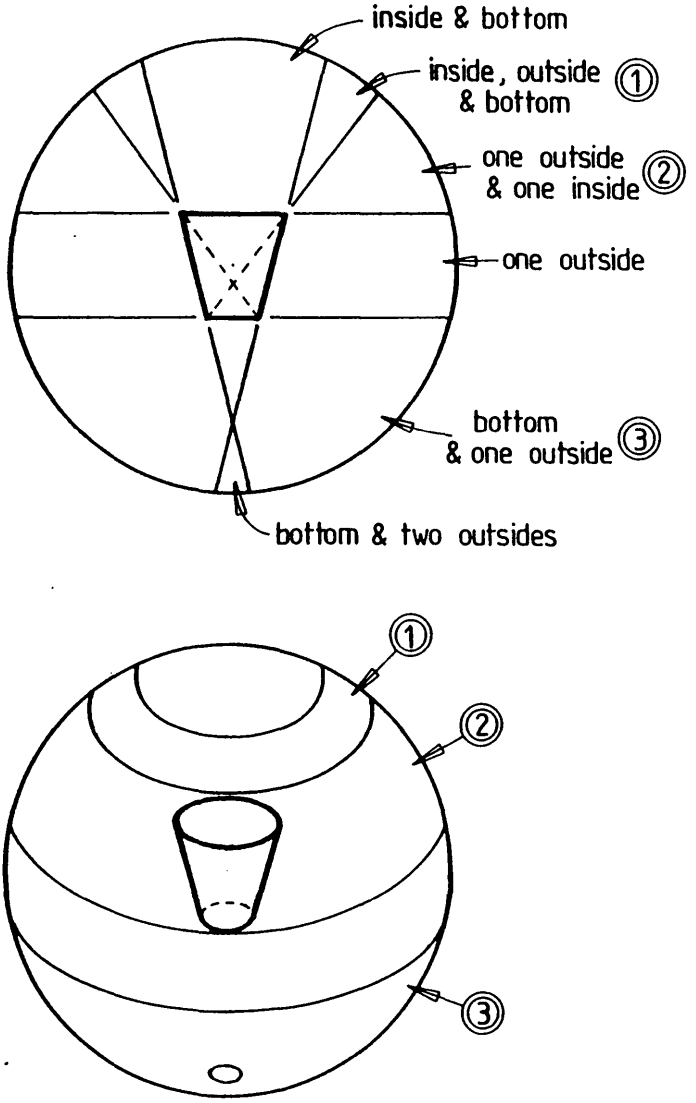


Figure 9–4: Distinct Viewing Regions for Trash Can

Figure 9–5: Spatial Configurations Invoke Models

Symbolic/Prototypical Objects Must Be Represented

Invocation is based on symbol matching, not geometry matching, representing the association of concepts. If there isn't a symbolic description for a structure, it can't be recognised. Prototypes arise because sometimes general models are needed. General properties of an object class may be more important for invocation (as compared to verification). An object's subfeatures are individuals but can often be represented by generic subfeatures having specific relationships. The subfeatures need to be distinctly characterized as components to represent their essential properties (identity and relationship to other subfeatures) and ignore inessential details. There are many types of wheels, but only the generic "wheel" is necessary when invoking most automobile models.

There is a Large Low-Level Vocabulary

There is a large vocabulary of low level, object independent and special intermediate shapes and configurations. If one looks at a Moore sculpture, such as

Figure 9–6: Sinusoid And Conjoined Semi-circles

the one used by Marr in his scale considerations ([MAR82], pg. 69), one vaguely "sees" it, but is ordinarily hard pressed to describe it in more than general terms, even via a sketch. Yet, the artist could undoubtedly sketch it accurately from any viewpoint. He probably had some conceptual unit for each of the different surface shapes, as well as a framework for their configuration. Another example arises from mathematical training. Figure 9–6 shows two superficially similar curves. Though a mathematically naive viewer could certainly distinguish the two, and possibly reproduce them, the viewer conversant with semi-circles and sinusoids is more likely to produce accurate descriptions and reproductions.

On the other hand, there must be a basic set of primitive, autonomously extracted features that underpin the learning of these new low-level descriptive terms.

The point is that there are conceptual associations made with shapes, and it is likely that one learns many. In the heart shape shown in figure 9–7, several distinct levels of shape features are isolated. So, even for simple objects, there is a complex set of sub-structures. If we generalise to 3D objects, we should again

expect to find many conceptual shapes and features, some of which are generic and some that are specific to a single or a few objects. The conclusion is that the conceptual space is dense with intermediate concepts between the raw image data and a 3D world object. This is presumed to be important to the invocation process, because of the filtering effect on recognitions. Data invoke low level visual concepts which invoke higher level concepts.

In part, there is a practical point to using the richer vocabulary. The symbols structure the description of an object, thus simplifying any direct model-data comparisons. This vocabulary increases efficiency through shared features. Further, if descriptions are sufficiently discriminating, a vision system may be able to accomplish most of its interpretation through only invocation with little or no model-directed investigation.

Invocation Is Incremental

The desirability of invoking a concept increases as more supporting evidence accumulates. Evidence can come from a variety of sources and these should be integrated. Complementary evidence should contribute to plausibility and conflicting evidence should detract from it. The process should degrade gracefully: less evidence should lower the desirability of invocation rather than prevent it totally.

The invocation process must continue even though evidence is missing because of occlusion or wrong descriptions. This requires a large variety of evidence for the object, so that having some of it missing will not cause a complete failure. This is probably a general requirement, as no primitive data analysis is likely to be complete or perfectly correct.

These factors suggest that invocation is mediated by a continuous plausibility of a concept explaining the data.

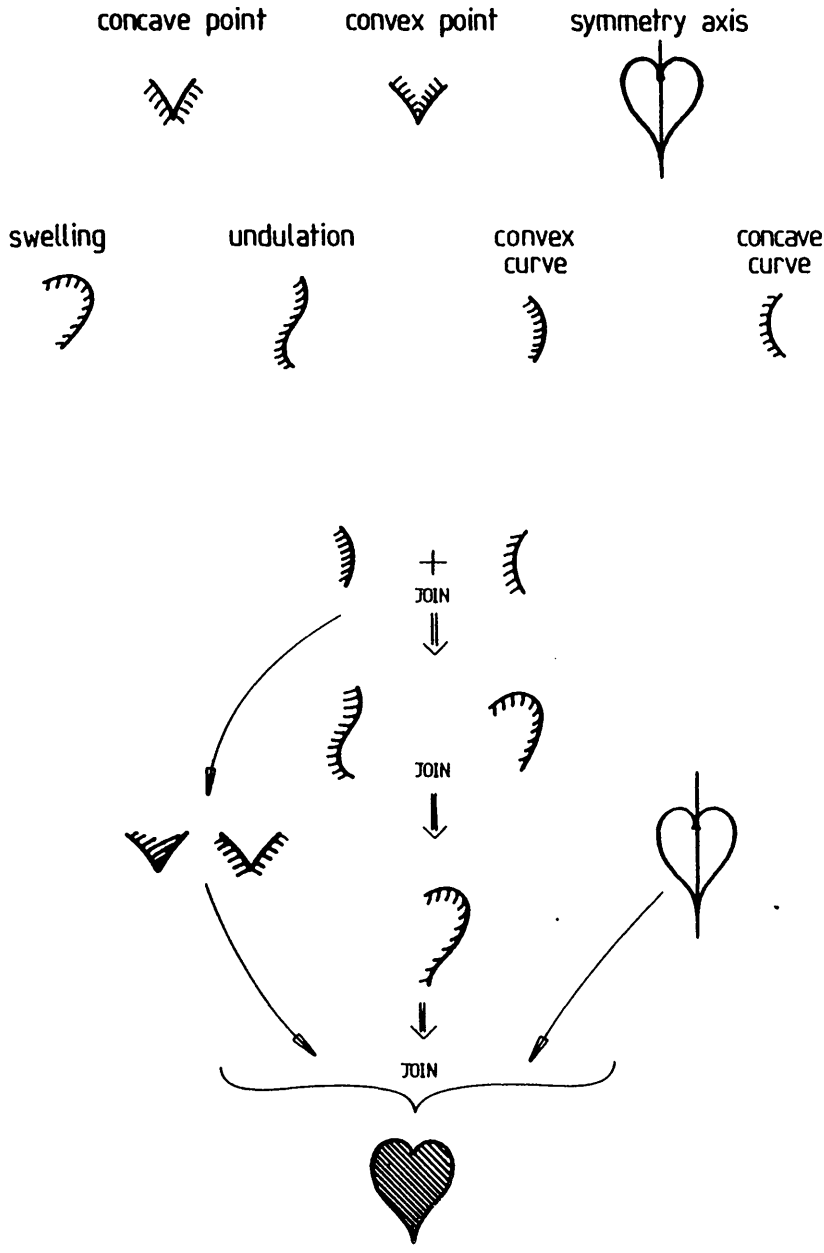


Figure 9–7: Heart Figure Structural Decomposition

Invocation Occurs In Image Contexts

There must be some context for grouping direct evidence together to suggest objects and for associating subcomponents so that indirect evidence accumulates in favor of the objects. The surface hypothesis (chapter 6) forms a natural focus for collecting direct evidence about that surface and the surface cluster (chapter 7) for direct evidence about solids. The surface cluster also groups surfaces belonging to the same object, so it is the context for accumulating plausibility from the surface subcomponents. Finally, the larger surface clusters aggregate volumes, and provide the context for relating solid subcomponent plausibilities.

Invocation Must Occur In Parallel

This is an implementation detail, but seems a necessary facet of a real solution given:

- that there are many contexts in an image which may invoke models and
- that there are many visual structures at both the low-level pre-object level and at the higher object-specific level, and somehow all need to be considered as potential candidates for explaining a set of data.

9.2 Theory: Evidence and Association

The previous section discussed the motivations for an invocation process and some factors impacting on its nature. This section formally proposes the process and details its structure based on the intuitions of the previous section.

The first consideration of invocation is from its externally viewed characteristics: its function and its input and output behavior. From figure 4–1, it can be seen that invocation occupies the place between the structure description and the hypothesis completion processes.

Model invocation calculates a plausibility measure representing the degree to which an object model explains an image structure. This measure lies in the range [-1,1]. When it is positive, the model is invocable.

The plausibility measure is a function of the model, the data context, the image properties (i.e. those in chapter 8), the desired properties, the model-to-model associations, current object hypotheses, and the plausibilities of all related model/data pairings.

A plausibility measure is used instead of direct indexing because:

1. many objects have similar features and plausibility measures similarity between models,
2. it allows weak evidence support from associated model/data pairings,
3. it supports accumulating unrelated evidence types, and
4. it provides graceful degradation as image descriptions fail because of noise, occlusion or algorithmic limits.

Invocation always takes place in a image context. This is because objects features are always connected and their features are always spatially nearby (for the types of objects considered in this thesis). The context defines where image data can come from and what structures can provide supporting evidence according to association type (more details below). For this research, the two types of contexts are the surface hypothesis (chapter 6) and the surface cluster (chapter 7), which localize evidence for model SURFACEs and ASSEMBLYs respectively.

More formally, the inputs to invocation are:

- A set $\{C_i\}$ of image contexts.
- A set $\{(d_j, v_{ij}, C_i)\}$ of image descriptions of type (d) with value (v) for the data in these contexts.

- A database $\{(t_i, M_j, M_k, w_{ijk})\}$ of model-to-model (M) associations of different types (t) with weights (w).

- A database $\{(M_i, \{(d_{ij}, l_{ij}, u_{ij}, w_{ij})\})\}$ of desired description constraints for each model, where d is the description type, l is the lower acceptable value for the description, u is the upper acceptable value and w is a weight.

- A set $\{(M_j, C_i, p_{ij})\}$ of model instances in contexts with plausibility values (p).

The output of invocation is a set $\{(M_j, C_i, p_{ij})\}$ of the plausibility measures for each model instance in each image context.

The invocation calculation evaluates the plausibility of model instances based on the compatibility of evidence. Some general properties of the plausibility measure can be given. There is some relative ranking between the same model in different contexts:

Model M_i is more plausible in context C_a than in context C_b if $p(M_i, C_a) > p(M_i, C_b)$.

Further, if model M_i implies model M_j, then in the same context C:

$$p(M_j, C) \geq p(M_i, C)$$

Unfortunately, not much can be said regarding the ranking of different models in the same context, because each has different evidence requirements.

Given the plausibility ranking, when should a model be invoked? Even if a model instance has the highest plausibility, it should not invoke the model if the absolute plausibility is low, as when analyzing an image with no identifiable objects in it. Two alternatives were considered. The first alternative is a minimum global threshold level of plausibility. This is arbitrary, but the invocation network described below strongly favors positive plausibilities as supporting and negative plausibilities as contradicting, so a threshold of zero makes good sense. This solution was adopted in this research. The second alternative is that

each type of object has its own threshold, which might also work well as the plausibility measures of separate types are incommensurate.

The basic structural unit of invocation is a model instance. These are $M_i(C_j)$, a given model in a given context. In theory, the models could be any conceptual element, but for the vision problem the instances are physical structures. For example, an instance could be a whole face, a subcomponent like a nose, a configuration of features, a particular type of feature (e.g. birth mark), a surface shape such as the swelling of a nostril, or a curve such as the profile of the nose. The model instance always occurs in a visual context and is an interpretation context's image data. This implies that each possible object type is considered for each context; fortunately, the context formulation has already achieved a reduction of information.

Invocation is based on suggestion, which arises from associations and evidence. The plausibility value of a hypothesis is a function of direct evidence arising from observed features and indirect evidence arising from hypotheses that have some association with the current one. A toroidal shape is direct evidence for a bicycle wheel, whereas a bicycle frame is indirect evidence.

The foundation of plausibility is direct evidence; if there were no direct evidence, associative conclusions would have no weight. Direct evidence is acquired by matching descriptions of image-based structures to model-based evidence requirements. These requirements define the notion that certain features are important in distinguishing the structure. Evidence for a side panel of the PUMA robot would include a surface region that was planar and had an expected length to width ratio and area, among other properties.

Evidence is cumulative: each new piece of valid evidence increases the plausibility of a structure. Evidence is also suggestive: each item of support is evaluated independently of the others and so does not actually confirm the identity of any structure. Section 9.2.1 describes the direct evidence calculation in greater detail.

Direct evidence arises in a second manner. When a model has been invoked,

it is subject to a model-directed hypothesis construction and verification process. If the process is successful, then the plausibility value for that object is set to 1.0. Alternatively, failure sets the plausibility to -1.0. These values are permanently recorded for the hypotheses and affect the other plausibilities in future invocations, either through associations or by also recording that identity in a larger context.

Indirect evidence comes from associations. Although there are many categories of association made between objects, we consider four distinguished types and a fifth, general association category. In the discussion of each of the association types below, three major aspects are considered: the type of association, the calculation of the association's invocation contribution, and the context from which the indirect evidence is taken.

The four distinguished types of association relating object A to object B are:

1. supertype: B is a more general class of object than A.

2. subtype: B is a more specific class of object than A.

3. supercomponent: B is an assembly of which A is a subcomponent.

4. subcomponent: B is a subcomponent of assembly A.

These four types have been made explicit for several reasons. Component relationships give strong circumstantial evidence for the presence of objects. An object necessarily requires most of its subcomponents for it to be considered that object, whereas the reverse does not hold. The presence of a car makes the presence of wheels plausible. The presence of automobile wheels also makes the presence of a car plausible, though the latter implication is weaker.

Generalisations and specialisations of concepts make explicit other associations. Generalisation links make available more general evidence. Specialisation links obtain suggestions from more specific objects. Specific types of cars have specific body shapes, but the general car has a general shape.

The different model hypotheses in the different contexts can thus be considered as nodes in a graph with the direct and indirect evidence relationships linking them. Thus, the whole invocation model base can be viewed as an associative semantic network, with the nodes as instances of models and the links as the association types. Direct evidence provides the raw plausibility values for a few of the nodes, and the other nodes acquire plausibility through association. It is hoped that only nodes corresponding to true scene entities achieve high plausibilities. The precise formulation of this network, with image registration, is discussed in section 9.3.

Another factor impacting on the identity of a structure is other potential identities. Because a structure seldom has more than one reasonable interpretation (except for generically related identities), highly plausible interpretations should inhibit other interpretations.

The final issue discussed in this section is evidence integration. Because there are six different evidence classes corresponding to the different direct and indirect evidence types discussed above, plus the type inhibition, the problem of how to compute a single plausibility value arises. The solution (section 9.2.8) treats the different evidence values as being on the same scale, but uses a function based on the types of evidence to integrate the values. The motivation for this is to use all the evidence, assuming evidence is always cumulative (as data errors, missing values, and object variations are alternate causes for weak evidence, as well as examining the wrong object).

An example is shown in figure 9–8, where a simple network is shown with the given association links ("g" denotes a generic relationship and "c" denotes a component relationship). The precise formulation of the calculations is given in later sections, and the point here is to introduce the character of the computation. Supposing there was direct evidence for there being a $< torus >$ and a $< vehicle >$ in the current context, the question is what is the invocation plausibility for the $< wheel >$. This value comes from integrating generic evidence from the $< torus >$ and component evidence from the $< car >$ and $< bike >$ and competing generic evidence from the $< polomint >$.

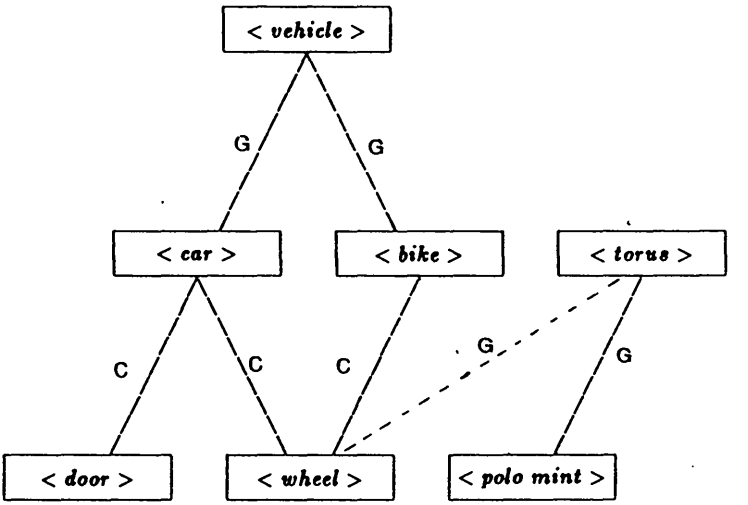


Figure 9–8: A Simple Invocation Network

In the following subsections, the theories for the direct evidence, association evidence, and accumulated plausibility computations are discussed in detail. The evaluation of the theory is discussed in section 9.4.

9.2.1 Direct Evidence

Direct evidence is calculated by matching structural descriptions to model-based evidence requirements. An example of a set of requirements would be that a particular surface is planar and meets all adjacent connected surfaces at right angles. The inputs are the evidence requirements for the structures, as detailed in chapter 5.

The question of what should be considered evidence is open. Here, evidence is based only on primitive descriptions, such as relative surface angles, rather than higher level descriptions, such as "rectangular". This decision is partly because "rectangular" is a distinct conceptual category, and, as such, would be included as a distinct generic element in the invocation network. It would then have a supertype relationship to the desired model.

These requirements and the availability of real descriptions motivated the description types given in chapter 8. While this thesis used only surface shape evidence, this is also the best place to introduce other evidence, like color, surface or reflectance texture, parallelisms, etc.

The context within which data is taken depends on the structure for which direct evidence is being calculated. If it is a model surface, then descriptions come from the corresponding surface hypothesis. If an assembly, then from the corresponding surface cluster.

Evidence requirements are defined in the model database in the form:

$$< min > \quad < \quad < evidence\_type > \quad < \quad < max > < weight >$$

This means that any data of the given evidence type should fall into the range (or ranges) given in the model database. The weight scales the contribution this evidence makes towards the total direct evidence value (described below).

Because the model database contains geometric models of the objects, it should be possible to automatically generate the evidence types used here, given heuristics for setting the ranges and weight values. However, in the current implementation, all these values were manually chosen and form an important part of the object model, as discussed in chapter 5. Appendix B shows the evidence constraints for the recognised objects.

Because supertypes of the object are also appropriate models to invoke, objects inherit constraints from their supertypes. This also has the practical effect that only additional evidence (i.e. refinements) need be given for more specialised objects.

Finally, we consider how the total evidence value is calculated from the individual pieces of evidence. The requirements on this computation are:

- Each piece of evidence should contribute to the total value.
- The contribution of a piece of evidence should be a function of its importance in uniquely determining the object.
- The contribution of a piece of evidence should be a function of the degree to which it meets its constraints.
- Negative evidence should inhibit more strongly than positive evidence support.
- Each piece of evidence should be considered only for the best fitting constraint.
- Not all constraints need evidence (e.g. because of occlusion, alternate viewpoints).
- All constraints for all generalisations of the model also apply here.
- Every description must meet a constraint, if any of the appropriate type exist.

- Not all description types are constrained (i.e. some properties are irrelevant).

Based on the degree of fit requirement, a function was designed to evaluate the evidence from a single description. The evidence falls off according to the distance from the normal value within a range and is at a minimum otherwise. The function is:

Let:

n = nominal value (from model) = $\frac{max+min}{2}$,

r = nominal range (from model) = $\frac{max-min}{2}$,

w = importance weight (from model),

d = data value

c = contribution

If: $|n - d| < r$

then: $c = w * (1 - 2 * \frac{|n-d|}{r})$

else: $c = -w$

Figure 9-9 illustrates the function. Other functions meet the requirements, but stronger requirements could not be found to define the function more precisely.

A description's evaluation is given by the constraint it fits best. This is because model-data correspondences have not been made yet, so the evaluations are undirected. For example, the evidence constraints on line length for a rectangle require all boundaries to be either 5 or 10 cm. long, so the boundaries from a 5 by 5 square would all meet the 5 cm. constraint. Two of the 5 cm. boundaries should be evaluated against the 10 cm. constraint, but this does not happen.

This ignores the fact that structures (e.g. lines) have identities arising from model correspondences and so only the appropriate constraints should be applied. Unfortunately, at this stage, feature identities are not established, so

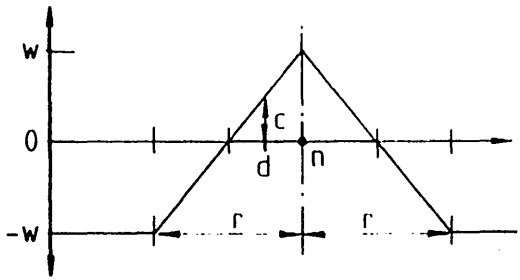


Figure 9-9: Data Evaluation Function

the corresponding constraints cannot be applied. Further, the identities are assigned only locally, so more global structure constraints, which could eliminate false identities, are not used. In any case, the point of invocation is suggestion, so any datum within any constraint range is contributory evidence. Correct models will have all constraints necessary for evaluation; incorrect models will acquire plausibility according to the closeness of match. Model-directed hypothesis construction (chapter 10) and verification (chapter 11) should remove most spurious invocations later.

Hence, the best of all weighted evaluations is selected as the datum's representative evaluation. This portion of the invocation network is illustrated in figure 9-10.

These best evaluations for individual descriptions then need to be integrated. Each piece of evidence is supposed to be correct, so weak evaluations should detract and strong evaluations support. Negative evidence should more strongly detract, because many models are likely to have some constraints satisfied by the image evidence, so any negative evidence should seriously weaken the invocation.

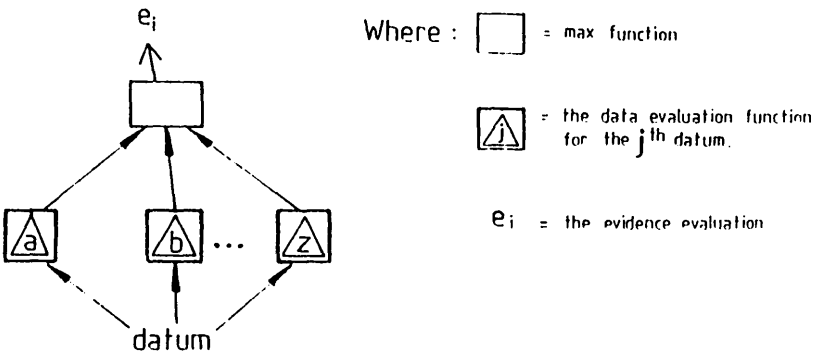


Figure 9–10: Best Evaluation Selection Network Fragment

At the same time, it should not cause immediate rejection, because the evidence may have arisen from partially obscured structures, or data errors. One function that meets these requirements is an average that incorporates negative evidence twice as strongly as positive evidence. (The requirements are not strong enough to implicate a single function.) The algorithm is:

Let:

e_i be the data evaluation for the i^{th} piece of evidence

w_i be the positive preference weight (from the network definition) for that evidence type

If: $e_i \geq 0$

Then: $a_i = e_i w_i$ and $b_i = w_i$

Else: $a_i = 2e_i w_i$ and $b_i = 2w_i$

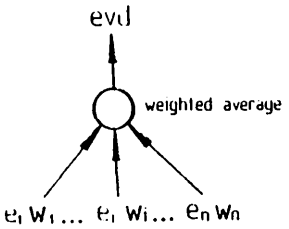


Figure 9-11: Network Fragment Integrating Direct Evidence

The final direct evidence evaluation is:

$$\frac{\sum a_i}{\sum b_i}$$

If no direct evidence is available, this computation is not applied.

This invocation network unit is shown in figure 9-11:

9.2.2 Supercomponent Associations

This association gives indirect evidence for the presence of a subobject, given the presence of an object. This seems like a natural association, though it is only suggestive because the subobject may not be visible (e.g. flaws or occlusion).

Though evidence typically flows from subcomponents to components, there are situations when the reverse is the primary effect. First, the component may have acquired direct evidence of its own, through solid constraints or having been recognized without all subcomponents, or have indirect evidence from generic or other relationships. Second, other subcomponents of the object may have been

found, which will contribute to the plausibility of the object, which will in turn contribute to that of those subcomponents not yet invoked.

One difficulty is that the presence of the supercomponent implies that *all* subcomponents are present (though not necessarily visible), but not that an image structure is any particular component. As the supercomponent evidence (during invocation) cannot discriminate between likely and unlikely subcomponents in its context, it supports all equally. This is because the computation is one of plausibility, not certainty. Weighting factors control the amount of support a structure gets, when integrated with other evidence. When support is given for the wrong identities, other evidence should contradict this and cancel it out.

Though the computation described below is speculative, there are several constraints derivable from the problem. They are:

1. The presence of an object makes the presence of its subcomponents plausible.

2. The more plausible the presence of the object, the more plausible the presence of the subobject.

3. At most one object is the true superobject of any subobject, though there may be many candidates.

4. The context of the superobject must contain the context of the subobject.

The formal definition of the supercomponent association computation is:

Given:

an object context C,

an object hypothesis $M(C)$ of type M,

a set of supercomponent relations $\{(M, S_i)\}$,
where S_i is the supercomponent type,

a set of contexts $\{C_j\}$ containing the context C, and

a set of supercomponent instances $X = \{(S_i(C_j), p_{ij})\}$
(i.e. a set of instances of type S_i, in context C_j,
with plausibility value p_{ij}),

Then, the supercomponent indirect evidence value associated with $M(C)$ is given by:

$$p_{spc}(M(C)) = max_X(p_{ij})$$

If no supercomponent evidence is available, this computation is not applied. There are other possible computations, as the constraints are not sufficient to lead to a unique solution. It is not known if all computations meeting the above constraints will solve the problem, but at different performance levels.

The $S_i(C_j)$ come from surface clusters nested by inclusion. The current context is also included because the supercomponent may not have been visually segmented from the subcomponent. Figure 9–12 shows the portion of the network associated with the accumulation of supercomponent evidence.

9.2.3 Subcomponent Associations

This association gives direct evidence for the presence of an object, given the presence of its subcomponents. Unlike the supercomponent associations, it is more than suggestive because the subcomponents are necessary features of the objects. Yet, it is not a complete implication because the subcomponents may be present in isolation, without the complete object, as with a bicycle wheel without the bicycle. Further, the implication is weak because the object requires specific relationships between its subcomponents, whereas the computation described below only requires the presence of the subcomponents. This would allow an unassembled bicycle to suggest the assembled one, which is reasonable. The point of invocation is to suggest potential models, not to constrain the analysis

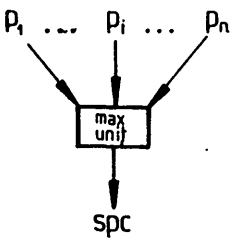


Figure 9–12: Supercomponent Evidence Integration Network Unit

sufficiently to produce verified results. For example, in figure 9–3 the structural relationships were not needed to invoke the face model.

This computation is defined by several constraints:

- The more subcomponents present, the more plausible the object's presence.
- Even if all subcomponents are present, this does not guarantee the presence of the object.
- Subcomponents are typically seen in viewpoint dependent groupings.
- The more plausible a subcomponent's presence, the more plausible is that of the object.
- The context of all subobjects must lie within the context of the object.

The computation is formalised below and looks for the most plausible candidate for each of the subcomponents, in the given context, and average their

contributions towards the plausibility of the object being seen from key viewpoints. The final plausibility is the best of the viewpoint plausibilities. Each of the individual contributions is weighted. The averaging of evidence arises because each subcomponent is assumed to give an independent contribution towards the whole object plausibility. [1] Because all subcomponents must lie within the same surface cluster as the object, the context of evidence collection is that of the hypothesis and all contained subcontexts.

The formal definition of the subcomponent association calculation is:

Given:

an object context C,

an object hypothesis $M(C)$ of type M,

a set of subcomponent relations $\{(M, S_i, w_i)\}$
where S_i is the subcomponent type
and w_i is the relationship weight,

sets of subcomponents $G_k = \{S_{ki}\}$ representing
groups of subcomponents visible from typical
viewpoints,

a set of subcontexts $\{C_j\}$ associated with
the context C, and

a set of subcomponent instances $X_i = \{(S_i(C_j), p_{ij})\}$
(i.e. a set of instances of type S_i, in

[1] By a common sense consistency notion, if an object has more than 50% of its subcomponents present, it is relatively certain to be present. Hence, the evidence function should probably be more than linear, though only a linear one was implemented.

context C_j, with plausibility value p_{ij}).

Let: $p_i = max_{X_i}(p_{ij})$,

If: $p_i > 0$

then

$$x_i = p_i w_i$$

$$y_i = w_i$$

else

$$x_i = 2p_i w_i$$

$$y_i = 2w_i$$

Then, the plausibility of a particular visibility subgroup G_k (over $S_i \epsilon G_k$) is:

$$v_k = \frac{\sum x_i}{\sum y_i}$$

And the final subcomponent indirect evidence value is:

$$p_{sbc}(M(C)) = max(v_k)$$

If no subcomponent evidence is available, this computation is not applied. The comment from section 9.2.2 regarding the uniqueness of the computation applies here as well.

The visibility subgroup calculation (v_k) is illustrated in figure 9–13. This calculation implements the constraint that features appear in groups, with only a subset of all possible features visible. Simultaneously, it also requires all features in the group to be present by accumulating the best evidence for each group member. This is because some viewpoints may share a single feature but seldom the entire group, which is the key structure here. The weight factors designate significance within the group, with larger weights emphasizing more important or significant features. The positive minus twice negative accumulation biases against groups with insufficient evidence.

Self-occlusion does not present a problem, because group membership does not include obscured components. Features obscured by unrelated objects can-

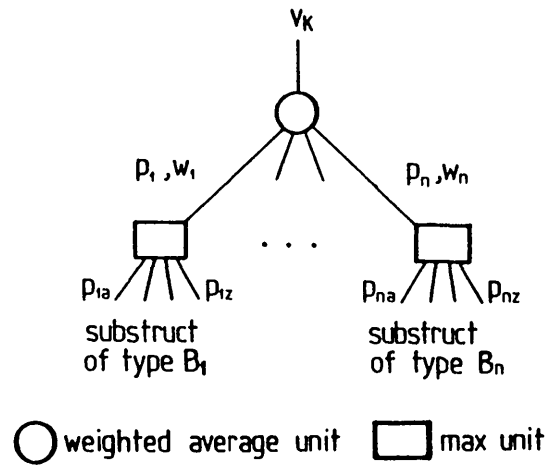


Figure 9–13: Visibility Subgroup Invocation Network Unit

not be anticipated easily, so some components may be missing from a group. If the evidence for other components is high enough, then invocation will proceed anyway. If not, the object is likely to be severely obscured, so recognition failure is to be expected. (Success might occur on a later invocation cycle if subcomponents or associated features are verified.)

Figure 9–14 illustrates how the visible subgroup plausibilities are integrated. The assumption is that if the object is present then there should be a subgroup whose features correspond to those visible in the scene. Hence, the subgroup with the highest plausibility should be selected as the visible configuration, and also give the final plausibility. If desired, the identity of the maximum visibility subgroup could be recorded, as it represents a rough orientation in viewer coordinates.

This computation does not explicitly account for the uniqueness of subcomponent identities. An image structure may contribute plausibility to two different subcomponents, or there may be multiple occurrences of the subcomponent present, and these features are not reflected in the network formulation. How-

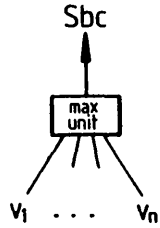


Figure 9–14: Invocation Network Unit Integrating Different Subgroups

ever, the identity inhibition contribution (section 9.2.7) helps force a single strong identity to any structure, so the first case should not occur often. The second case is not currently handled.

Because the visibility subgroups have a definable visible aspect, they should perhaps be defined as distinct objects, such as "the back of a head", or a "face", with respect to the whole head structure. This line of reasoning would replace the two-tiered computation here with two separate ones. The first computes the identity for the distinct views, and the second for a new relationship something like aspect, projection or homomorphism. Under this structure, the different aspects need not be independent. This generalisation might also be needed to extend visual invocation to the general cognition case, as suggested in section 9.1.

The visibility group could also be extended to include other features, such as color, texture, etc. Following this line might lead to partially unifying this computation with the direct evidence computation (section 9.2.1).

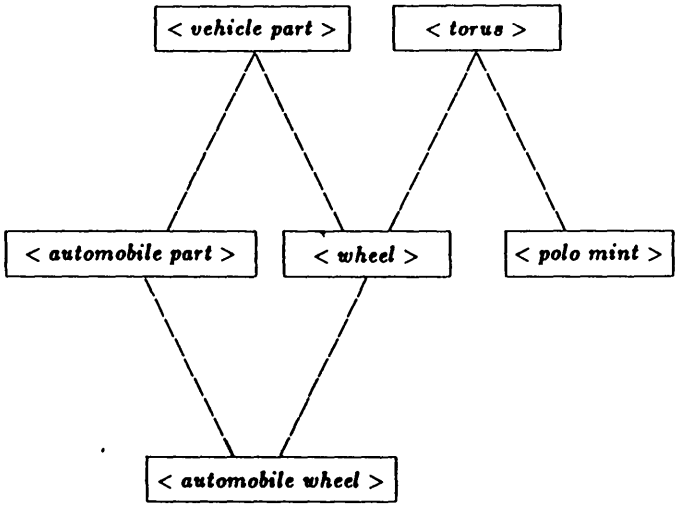


Figure 9–15: A Simple Type Hierarchy

9.2.4 Supertype Associations

This association gives indirect evidence for the presence of an object of type M, given the presence of an object of supertype S. For example, evidence for the object being a wide-bodied aircraft lends some support for the possibility of it being a DC-10. The notion of supertype is not being used rigorously here – the intuitive notion is that of a category generalization along arbitrary lines. Hence, a type may have several potential generalizations, as in <wheel> or <automobile part> in figure 9–15. Further, type generalization is weak and does not require that all constraints on the generalization are satisfied by the specialization, as a wheel is not strictly a torus. No formal definition of the relationship is proposed, and a practical definition will ultimately be required.

Supertypes provide circumstantial, rather than direct, evidence as the presence of the <torus> alone does not provide serious evidence for the <wheel> being present. If the object had both strong <torus> and <vehicle part> evidence, the implication should be stronger. However, if the object had strong <torus> and weak <vehicle part> evidence, then it would be less plausible for it to be a <wheel>. Because the supertype is a generalization, its plausibility must always be at least as great as that of the type. Hence, the evidence for a type can be at most the minimum of the evidence for the supertypes.

Supertypes do not always linearize, because of the object definition based on constraints. For example, two unrelated generalizations of "red delicious apple" are "red spheroid" and "apple". Class formulation requires the first to lie in the intersection of the two superclasses, and hence the constraints for the superclasses must be subsets of the specialization.

The constraints that help specify the supertype evidence computation are:

- The presence of a supertype increases the plausibility of the subtype being present.
- The more plausible the supertype, the more plausible the type.
- The plausibility of a type is less than that of the minimum of its supertypes.
- The context of the supertype is that of the type.

These constraints lead to the following formal definition of the supertype association computation:

Given:

an object context C,

an object hypothesis $M(C)$ of type M

a set of supertype relations $\{(M, S_i)\}$

(where S_i is the supertype),

a set of supertype instances $X = \{(S_i(C), p_i)\}$
(i.e. a set of instances of type S_i, in context C,
with plausibility value p_i),

Then, the supertype indirect evidence value associated with $M(C)$ (over $S_i \epsilon X$) is:

$$p_{spt}(M(C)) = min_X(p_i))$$

If no supertype evidence is available, this computation is not applied.

This estimates the plausibility of type M given the plausibilities of its supertypes S as assessed in the same object context. Since all supertypes are imperative for the type, the evidence for the type is the minimum plausibility of any supertypes. The portion of the network associated with this evidence is shown in figure 9–16.

9.2.5 Subtype Association

This association gives indirect evidence for the presence of an object of type M, given the presence of an object of subtype S. As in the previous section, the notion of subtype is that of a specialization, as in an <automobile part> being specialised from <vehicle part>. Hence, a type may have several specializations, as in figure 9–15. Here, the implication is a necessary one, because an object of a given subtype is necessarily an object of the type. Hence, the plausibility of the supertype must not be less than that of the subtype. Subtypes may be related, as in the example in figure 9–15, where an <automobile wheel> is a object type that is a subtype of both subtypes of <vehicle part>. If there were multiple subtypes, then the type's plausibility should be at least the maximum of these.

The constraints that specify the subtype association calculation are:

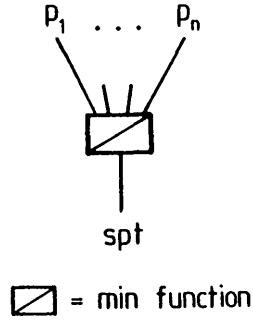


Figure 9–16: Supertype Evidence Integration Network Fragment

- The more plausible the subtype, the more plausible the type.
- The plausibility of the type is at least the maximum of the subtypes.
- The context of the subtypes is the context of the types.

These constraints lead to the following formal definition of the subtype association computation:

Given:

an object context C,

an object hypothesis $M(C)$ of type M,

a set of subtype relations $\{(M, S_i)\}$
(where S_i is a subtype of M),

a set of subtype instances $X = \{(S_i(C), p_i)\}$

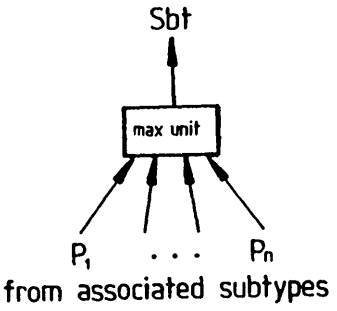


Figure 9–17: Subtype Evidence Integration Network Fragment

(i.e. a set of instances of type S_i, in context C,
with plausibility value p_i)

Then, the subtype indirect evidence value is:

$$p_{sbt}(M(C)) = max_X(p_i)$$

If no subtype evidence is available, this computation is not applied.

This estimates the plausibility of type M given the plausibilities of its subtypes S as assessed in the same object context. Since all subtypes imply the type, the evidence for the type is the maximum of the subtypes. There is no evidence weighting, as the implication is necessary. As types and subtypes refer to the same object, the contexts must be identical. Figure 9–17 shows the invocation network unit for this evidence.

9.2.6 General Associations

This association gives indirect evidence for the presence of an object of type *M* given the presence of an object of arbitrary type S. This is not a structural or type hierarchy based association; it is included as an "other associations" category. An association of this type might be: "the presence of a desk makes the presence of a chair plausible". This form of implication is weak, but allows many forms of peripheral evidence.

Association is not a commutative process, so individual connections need to be made, if desired, for each direction. The presence of a man makes the presence of a pair of trousers likely, whereas the reverse is not true.

The previous evidence types had clearly specified contexts from which evidence came, but this type does not. Generally associated objects could be anywhere in the scene, so all nodes of the desired type give support. This requires a more substantial network commitment than for the other types.

There are only weak constraints for this type of association:

- The presence of an associated object increases the plausibility of the required object.
- The more plausible the associated object, the more plausible the object.
- The weight of an association expresses the expectation that the desired object is present, given that the associated object is present.
- The plausibility of the object should be at least the maximum of the weighted plausibilities of its associations.
- The context of the association is the whole image.

These constraints lead to the following formal definition of the general association computation:

Given:

an object context C,

an object hypothesis $M(C)$ of type M,

a set of general association relations $\{(M, S_i, w_i)\}$
(where S_i is the associated object type),

a set of all contexts $\{C_j\}$,

a set of association instances $X = \{(S_i(C_j), p_{ij})\}$
(i.e. a set of instances of type S_i, in context C_j,
with plausibility value p_{ij}),

Then, the association indirect evidence value associated with $M(C)$ is:

$$p_{ass}(M(C)) = max_X(p_{ij}w_i)$$

If no association evidence is available, this computation is not applied.

Since all associates imply the object, the evidence for the object is the maximum of the associates. The evidence weighting stresses the importance of the association. Unfortunately, the formalisation does not handle multiple supporting evidence well. This point is partially addressed in section 9.4. Figure 9–18 shows the invocation network unit for integrating association evidence.

9.2.7 Identity Inhibition

A structure seldom has more than one likely identity, unless the identities are generically related (i.e. a DC-10 can also be a wide-bodied aircraft but seldom a banana). Hence, an identity should be inhibited by other unrelated identities having high plausibilities in the same context. A second source of inhibition comes from the same identity in subcontexts, to force invocation to occur only

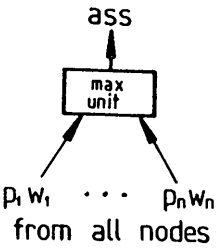


Figure 9–18: Association Evidence Invocation Network Fragment

in the smallest containing context. The key questions are how to quantize the amount of inhibition and how to integrate this inhibition with the other evidence types.

For simplicity, the inhibition was assumed to result in a plausibility value like those discussed in the previous sections. It can then be integrated with the other evidence types, as discussed in section 9.2.8. An advantage to this method is that it still allows for alternative interpretations, as in the ambiguous duck/rabbit figure (e.g. [ARB79]), when evidence for each is high enough.

The constraints on the inhibition computation are:

- Inhibition can only come from generically unrelated types or the same type in contained contexts.
- Only positive evidence for other identities inhibits.
- The inhibition should be proportional to the plausibility of the competing identity.

- The inhibition should come from the strongest competition.

- The context of inhibition is the current context for unrelated identities and all subcontexts for the same identity.

These constraints lead to the following definition of the inhibition computation:

Given:

an object context C,

an object hypothesis $M(C)$ of type M,

a set $\{S_i\}$ of all identities generically unrelated to M,

a set of subcontexts $\{C_j\}$ of context C,

a set $X = \{(S_i(C), p_i), (M(C_j), p_j)\}$
of plausibilities p for the identities S_i in context C
and the identity M in subcontexts C_j.

Let:

$$v = max_X(p_i)$$

Then, the inhibition evidence associated with $M(C)$ is:

$$p_{inh}(M(C)) = -v \text{ if } v > 0 \text{ else no inhibition}$$

This computation also gives no inhibition if no competing identities exist.

Inhibition is treated as negative evidence, as a function of the maximum competing positive evidence. If several identities have roughly equal plausibilities, then inhibition drives their plausibilities down, but still leaves them roughly

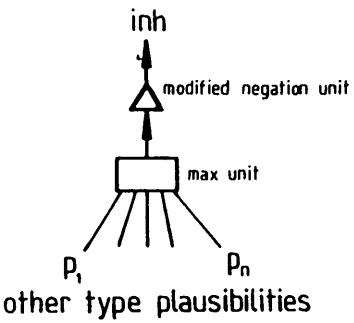


Figure 9–19: Inhibition Invocation Network Fragment

equal. A single strong identity would severely inhibit all other identities. Figure 9–19 shows the invocation network unit for computing the inhibition evidence.

9.2.8 Evidence Integration

There are seven evidence types, as discussed in previous sections, and a single integrated plausibility value needs to be computed from them. All values are assumed to be on the same scale so this simplifies the considerations. This may not be a reasonable assumption, but then weighting factors could be added that adjust the relative importance of the individual results in the final calculation.

Some general constraints the computation should meet are:

- Directly related evidence (direct, subcomponent and subtype) should have greater weight.

- Other indirect evidence should be incremental, but not overwhelmingly so.

- Only types with evidence are used (i.e., some of the evidence types may not exist, and so should be ignored).

- If there is no direct, subtype or subcomponent evidence, then evidence integration produces no result.

More specific constraints are:

- Direct and subcomponent evidence are complementary in that they both give explicit evidence for the object. If one is weak and the other strong, then the weak evidence should be followed, because the object must have both sets of properties.

- If supercomponent evidence is strong, then this gives added support for a structure being a subcomponent. Weak supercomponent evidence has no effect, because the subcomponent could be there by itself.

- As subtypes imply types, the plausibility of a type must be at least that of the subtype.

- As types imply supertypes, the plausibility of a type must be at most that of the supertype.

- Strong association evidence supports the possibility of an object being present. Weak association has no effect, because the object could be there by itself.

- If other identities are competing, they inhibit the plausibility.

Based on these constraints, the following computation has been designed:

Let:

$$e_{dir}, e_{subt}, e_{supt}, e_{subc}, e_{supc}, e_{ass}, e_{inh}$$

be the seven evidence values.

Then:

$$v_1 = min(e_{dir}, e_{subc})$$

if $e_{supc} > 0$

then $v_2 = v_1 + c_{supc} * e_{supc}$ $(c_{supc} = 0.1)$

else $v_2 = v_1$

if $e_{ass} > 0$

then $v_3 = v_2 + c_{ass} * e_{ass}$ $(c_{ass} = 0.1)$

else $v_3 = v_2$

if $e_{inh} > 0$

then $v_4 = v_3 + c_{inh} * e_{inh}$ $(c_{inh} = 0.25)$

else $v_4 = v_3$

Finally, the integrated plausibility value p is:

$$p = min(max(v_4, e_{subt}, -1.0), e_{supt}, 1.0)$$

The invocation unit executing this function is shown in figure 9-20.

9.2.9 Examples of Invocation

Several examples are now presented. Suppose interest is in the plausibility of the trash can outer surface seen as region 9 in image 1 of appendix A. The description process produces the following results:

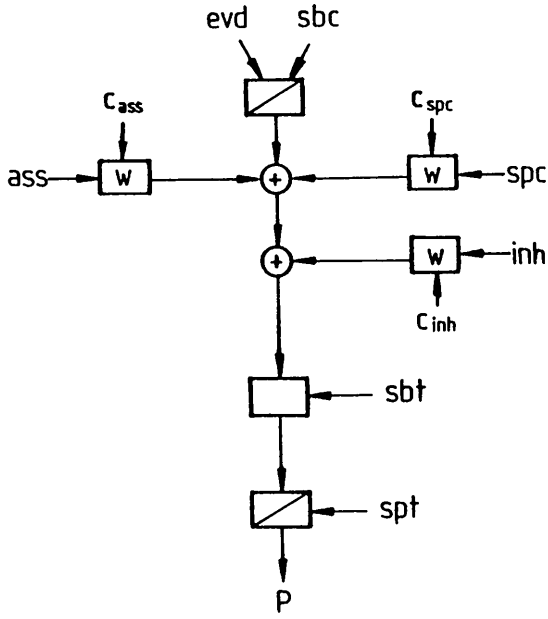
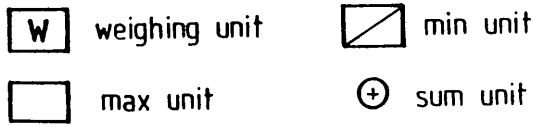


Figure 9–20: Evidence Integration Invocation Network Fragment



description	values
maximum surface curvature	0.081
minimum surface curvature	0.0
relative size	0.93
absolute size	1085
elongation	1.84
boundary relative orientation	1.64, 1.45, 1.45, 1.73
parallel boundaries	2
boundary curve length	32.7, 25.3, 30.1, 28.0
boundary curvature	0.054, 0.012, 0.083, 0.0

The direct evidence computation is then performed, based on the evidence constraints, as given in the model (from appendix B):

DESCRIPTION	LOW	HIGH	WEIGHT
solid surface angle	2.97	3.3	0.5
	4.48	4.68	0.5
maximum curvature	0.058	0.098	0.5
minimum curvature	-0.003	0.015	0.5
relative size	0.40	0.99	0.5
absolute size	980.0	1140.0	0.5
elongation	1.4	2.0	0.5
boundary relative orientation	1.3	1.85	0.5
parallel boundaries	1	3	0.3
boundary curve length	19.0	39.0	0.5
	25.0	45.0	0.5
boundary curvature	0.05	0.11	0.5
	-0.003	0.003	0.5

This results in a direct evidence value of 0.47, as computed by the process described in section 9.2.1. Assuming invocation has run to convergence, there are other evidence values. There are no subtypes, supertypes, subcomponents or associations, so their evidence contribution is not included. The supercomponent plausibility is 0.34 because the surface belongs to the trashcan ASSEMBLY. The maximum of the other surface plausibility values for non-generically related identities is 0.33 (for the trash can inner surface model), so this causes some inhibition.

These evidence values are now integrated according to the computation given in this section, to give the final plausibility value for the surface as 0.42. As this is positive, the trash can outer surface model will be invoked for this region.

Figures 9–21 and 9–22 show another example of evidence integration. Figure 9–21 shows a simple scene of a trash can with two surfaces exposed. For each of the two surfaces (A and B), three possible identities obtain: trash can inner surface, trash can outer surface and trash can bottom. For the surface cluster consisting of both surfaces, two possible identities are relevant: open cylinder and trash can. Figure 9–22 shows the portion of the invocation network associated with these identities, as related to the calculation of the plausibility that the surface cluster is the trash can. The six bubbles at the bottom of the network are nodes representing the three possible identities for each of the two surfaces. The two bubbles at the top represent the two possible identities for the surface cluster. We now show how the trash can's plausibility arises from the influence of the other nodes.

This network fragment records only subcomponent, supertype and inhibition linkages, and others are presumed to be non-existent for this identity. The lower part of the network shows the computation for the subcomponent evidence. The model defines three subcomponents for the trash can:

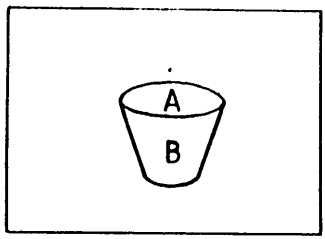


Figure 9–21: Trash Can Scene

```
SUBCOMPONENT OF trashcan IS tcanoutf 0.90;
SUBCOMPONENT OF trashcan IS tcaninf 0.90;
SUBCOMPONENT OF trashcan IS tcanbot 0.90;
```

and these are organized into these visibility groups:

```
SUBCGRP OF trashcan = tcanoutf tcaninf;
SUBCGRP OF trashcan = tcanoutf tcanbot;
SUBCGRP OF trashcan = tcaninf tcanbot tcanoutf;
```

One round averaging function unit corresponds to each of the subcomponent groups, and they take their inputs from the square maximum units below,

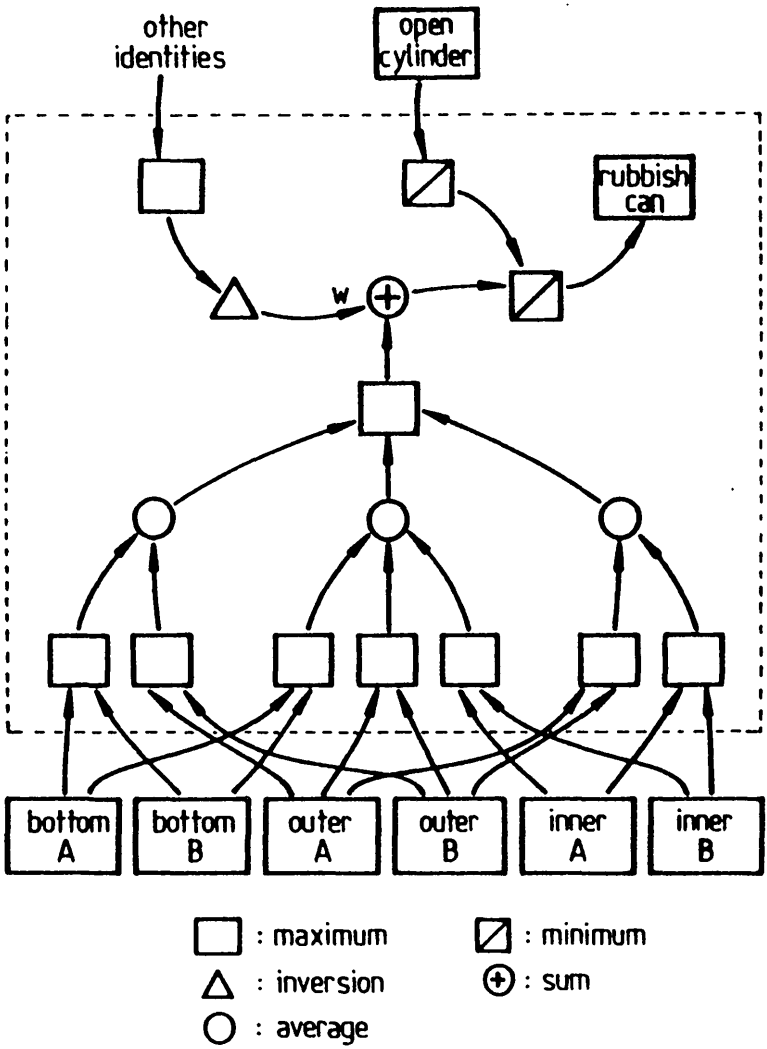


Figure 9–22: Trash Can Plausibility Calculation Fragment

which select the best structure for the required identity. The square unit above the round units picks the best visibility group to give the final subcomponent plausibility (as defined in section 9.2.3).

The bubble at the top is the supertype generalization of the trash can. Its relationship was defined in the model by:

```
SUPERTYPE OF trashcan IS opencyl 1.0;
```

The weight of 1.0 is not used here. Its plausibility contribution calculation comes through the square-bar unit (as defined in section 9.2.4). Here, there is only one input so the calculation is trivial.

The competing identities for the structure come in to the triangular inhibition unit from the left hand side of the diagram.

Finally, the units at the center integrate the three evidence types to give the plausibility for the trash can unit. Note that this is only a portion of the complete network, because the trash can node also influences the plausibility of all related nodes, and none of the outputs to other nodes is shown.

9.3 Implementation in a Visual Context

The previous section described the units in the invocation network and how they are interconnected. This section reports on how the whole network is constructed for scene analysis, and how it is evaluated.

Three key questions on network formation are:

1. What are the nodes in the network?

2. What are the principles that determine which nodes connect to each other?

3. How are the structures related to the image or scene?

Two types of nodes make up the network: surface identity nodes and solid identity nodes (edge features were not implemented). One solid identity node is created for each surface cluster/model ASSEMBLY pair and similarly for each surface hypothesis/model SURFACE pair. Each node can be placed in a plane of just its own type, with positions spatially corresponding to the 2D mosaic of image structures (e.g. one unit can be associated with each surface region).

The visual contexts were defined as being surfaces for surface identities and surface clusters for solid identities, and these contexts determine which nodes are linked. Thinking of our 2D metaphor for the layout of the network in registration with the image, only nodes within the boundaries of the context are linkable. Hence, an ASSEMBLY can only use surface or subcomponent evidence from nodes associated within its surface cluster.

Figure 9–23 shows this network organization for the example of figures 9–21 and 9–22. Here, the boundaries of surface cluster D show the proper context for the connections to trash can node T_D which includes the nodes $C_D, I_A, I_B, O_A, O_B, B_A, B_B$. These correspond to the named bubbles shown in figure 9–22.

Among the potential connections, the model relationships (as given in chapter 5) specify which node types should be connected, and all nodes of these types with the contexts are connected.

This regular image-related structure can be closely mapped onto a regular 3D parallel machine. Assume that each of the 2D planes has many individual processors laid out in a regular 2D array. Then, assume all processors are cross-connected to all neighbors except for where the connection crosses a context boundary. These context boundaries partition the 2D array into groups of processors. Then, if the connections cause all processors to compute the same value, then each connected group is logically equivalent to a image structure/model identity node (see figure 9–22). The image context boundaries gate the horizontal interprocessor connections, so *the same network can be used for different images.* The vertical connections are determined completely by the model, and so all individual trash can processors would connect vertically to the corresponding generic open cylinder processors. By the equivalent connections

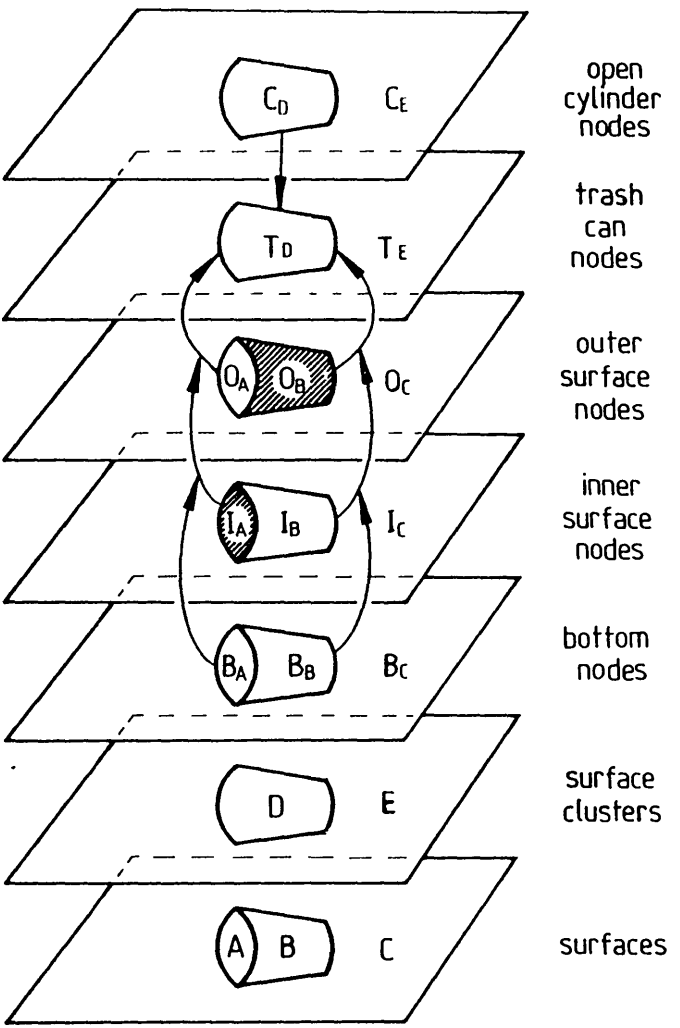


Figure 9–23: Spatial Registration and Context in Invocation

inside a context, all processors of the same type receive and output the same values.

One unresolved problem with this scheme is how to integrate evidence from different subcontexts, as in the subcomponent evidence from contexts A and B in figure 9–23 must be integrated into context D, because not all processors in D link to a processor in A, or B.

A second problem concerns how many surface cluster planes to allow, as extended surface clusters are needed to integrate evidence from self-obscuring objects (chapter 7). This implies that each ASSEMBLY might have several planes, one for each surface cluster level. This seems excessive, and since the extended surface cluster problem itself is also unsolved, this problem must await future developments.

This highly parallel invocation structure could use a separate processor for each image pixel/object model pair, with neighbor links according to the context constraints defined in section 9.2. Because most of the computations are numerical, the computation could be implemented on a parallel analog machine. Positive and negative plausibilities could be implemented by activity on paired complementary connections. This invocation machine could have a pyramid-like architecture, like those proposed for more general image processing (e.g. [TAN78]).

For the implementation evaluated in this thesis, networks had one node for each identity/context pair, the regular 3D structure was implicit only in the problem, and only the internode linkages determined by the models and contexts were made (i.e. no explicit cross-connection disabling by context boundaries).

The point of invocation is to reduce the computation involved in the data-to-model matching process. This has been partially achieved by basing invocation on propagated plausibility values, so the computation has been reduced from a detailed object comparison to evidence accumulation. Unfortunately, virtually every object model still needs to be considered for each image structure, albeit in a simplified manner.

On the other hand, the data to model comparison computation has now been regularized. As a result, it is now amenable to large scale parallel processing. The computation is similar to that of relaxation algorithms ([ROS78]): there are multiple entities (image structures), several possible labels for the entities (object models) with probability values (plausibilities), input data (direct evidence) and relationships (indirect evidence) that must hold between the data, entities and labels. The goal is to assign a weight value for each label of each entity, such that all relationships still hold true. Previous researchers have tended to use relaxation algorithms to adjust data to conform to some notion of consistency, and then assume that the label weights denote some measure of certainty. In this research, the label values are used only to suggest plausibility, and certainty is determined later.

The ideal computation has the network converge as new descriptions are computed, assuming the invocation process executes independently of the data description process. The previous final state would be a reasonable initial estimate for the new solution, so convergence should be rapid. Further, when there is enough data to cause a plausibility to go above the invocation threshold, and that invocation leads to a successful recognition, then description of that structure could cease.

The parallel formulation is just speculation, unfortunately. The thesis implementation computes all possible descriptions initially. Then, plausibilities are iteratively computed for the entire network. Each iteration computes the plausibility for each node using the values from previous iterations until convergence is reached. On convergence, nodes with positive plausibilities are invoked for model-directed processing (chapter 10). Invocations are ordered from simple-to-complex (and then high to low plausibility) to ensure that subcomponents are identified for use in making higher level component hypotheses (chapter 10). This ordering seems contrary to the goal of invoking the most plausible models first. If some subcomponents were missing, the model-directed processing could propose these directly. This heterarchical structure was not investigated.

Because an object may appear in several nested surface clusters, it makes

little sense to invoke it in all of these after it has been successfully found in one. Further, a smaller surface cluster containing a subcomponent may acquire weak plausibility for containing the whole object. These too should not cause invocation. The inhibition formulation partly controls this, but one active measure was also needed. After an object hypothesis is successfully verified (chapters 10 and 11), the hypothesis is stored associated with the smallest surface cluster completely containing the object. All surface clusters containing or contained by this cluster then have their plausibility for the model set to -1 and a flag set to say that this model has been considered.

As the plausibility calculation involves feedback, the network tends to oscillate. An averaging computation was used to dampen the oscillations:

Let:

$p_{ij}(t)$ = plausibility of the i^{th} image structure having the j^{th} identity in cycle t

$f_{ij}(t)$ = the new plausibility for node ij calculated from all other appropriate node plausibilities at cycle t.

Then:

$$p_{ij}(t+1) = (p_{ij}(t) + f_{ij}(t))/2$$

This algorithm is not guaranteed to converge, but worked with various network weightings and input data values.

9.4 The Evaluation of Invocation

This section evaluates the theory described in the previous sections. There are three main topics: evaluation criteria, performance and critical discussion.

Evaluation Criteria

The properties that invocation should have are:

1. Correct models are always invoked, and only in the correct context.

2. No matter how large the model base, the only false hypotheses invoked are those "similar" to the true ones. Here, "similar" means having several properties in common.

3. The invocations are suggestive.

4. Invocation integrates multiple types of evidence.

This subsection presents the evidence that the proposed theory has these properties.

The proposed theory is not suitable for explicit experimentation. General mathematical or computational analysis would probably not contribute much either, as:

1. The performance depends on the particular network used, and there are few constraints on this.

2. The network executes a complicated, non-linear computation, and is thus hard to characterize.

3. No valid statistics are available for the performance of structure description (chapter 8) on general position, unobscured structures.

4. It is not possible to characterize the scenes sufficiently well to predict typical structure occlusions.

5. Little information is available to assess performance of the structure description on partially obscured structures.

6. Abstract invocation ordering or convergence rates are not interesting, only that the appropriate hypotheses will be invoked and that the network converges in a few cycles.

Hence, formal analysis is not likely to contribute much to assessing the theory in this chapter.

Three minor analytic results have been found:

(1) If all direct, subcomponent and generic evidence is perfect, then the correct model is always invoked. This is equivalent to saying that the object has the correct identity and all properties are measured correctly. Assuming the other three evidence types are totally contradictory, then the evidence integration calculation gives (section 9.2.8):

Let:

$$e_{dir}(1.0), e_{subt}(1.0), e_{supt}(1.0), e_{subc}(1.0), e_{supc}(-1.0),$$
$$e_{ass}(-1.0), e_{inh}(-1.0)$$

be the seven evidence values

Then:

$$v_1 = min(e_{dir}, e_{subc}) = 1.0$$

$$v_2 = v_1 = 1.0$$

$$v_3 = v_2 = 1.0$$

$$v_4 = v_3 + c_{inh} * e_{inh} = 0.75 \qquad (c_{inh} = 0.25)$$

and the integrated plausibility value p is:

$$p = min(max(v_4, e_{subt}, -1.0), e_{supt}, 1.0) = 1.0$$

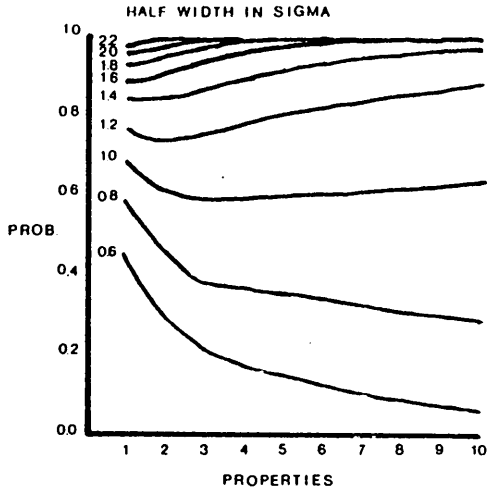


Figure 9–24: Probability of Positive Direct Evidence Versus Properties

(2) Assume N independent properties are measured as direct evidence for a structure, and all are equally weighted. Then, the probability that the direct evidence evaluation is greater than zero is shown in figure 9–24, assuming all constraint ranges have the widths given, and the data values are normally distributed. The point is to estimate how many properties are needed and what the constraint ranges on a property should be, to ensure that the direct evidence almost always supports the correct identity. The curves shows that if at least 6 gaussian distributed properties are used, each with constraint width at least 1.6 standard deviations, then there is a probability of 0.98 for positive direct evidence. These results were found by simulation.

(3) There are cases when the network oscillates instead of converging.

Let:

$f(i)$ be the state of a node in the network at time i

$g(i)$ be the new calculated value based on the whole network at time i

The updating function is:

$$f(i+1) = (f(i) + g(i))/2$$

One oscillation occurs if:

$$f(i+2) = f(i)$$

which implies

$$g(i+2) = g(i).$$

One state where this oscillation occurs is:

$$f(i) = a - b \qquad g(i) = a + 3b$$

$$f(i+1) = a + b \qquad g(i+1) = a - 3b$$

This has not been a problem in practice, though, as the network is complicated and it may be likely that such states are seldom encountered. Because of the use of the non-linear max and min operators, there may be cases where the network oscillates between states choosing different inputs as the maximums. To avoid occasional problems, computation is limited to 20 cycles. All oscillations observed during development involved nodes with negative (i.e. non-invocating) plausibilities, so terminating the computation caused no bad side effects.

The solution proposed in this chapter is reasonable because the proposed computation accounts for and integrates the major evidence types. The surface cluster contexts focus attention to assembly identities (and surface contexts to surface identities), the object types denote natural conceptual categories, and the different association links structure the paths of suggestion. The associations based on generic and component relationships is a strength. The continuous plausibility value formulation is a less certain approach, because *discrete* discrimination nets or *discrete* associative processes have implemented alternative solutions (chapter 2). But, these have difficulties with contradiction and integrating alternative evidence.

While the structure seems reasonable, there is the question of whether the proposed algorithms are as well. For each computation, some natural constraints

were proposed as specification criteria. But, as discussed, there were never enough constraints to uniquely determine the computation. The hope is that the variations in algorithms that this allows result only in slightly different performance levels.

These arguments support the claim that the third and fourth properties (from above) are held by the invocation network. The mathematical results suggest that the first holds if the data is well-behaved (though the likelihood of this is hard to assess). The second property is not easily assessed without a formal definition of "similar".

Other than the three mathematical results above, no simple criterion has been found that ensures that correct invocations are likely and false invocations are not. Neither, because of the small data set size, is there any statistical measure of performance. So, a performance demonstration is presented instead. This gives invocation position data from both test images, with a discussion of failures, and will show that, for the images analyzed, invocation is effective and robust.

Test Image Performance

The invocation process was run on the two images shown in appendix A, using the full model base given in appendix B. Several results from these are presented below.

In table 9–3, there are the invocation plausibilities for the example image 1 (all values are times 100). The values shown were formed as a result of combining the plausibilities of the evidence types, whose values are shown in tables 9–4 to 9–7, according to the computation described in section 9.2.8. The object types are listed across the top of the table and the image structures along the side. These correspond to the models in the model base (table 9–1) and the surface clusters of test image 1 (table 9–2). Only the object level plausibilities are displayed here. (No generic associations were modeled, and ASSEMBLYs had no direct evidence. Hence, these evidence types are not shown.)

Table 9–1: Model Correspondences for Data Tables

HORIZONTAL INDEX	MODEL
0	hand
1	lowerarm
2	upperarm
3	upperasm
4	robshldbd
5	robshldsobj
6	robshould
7	link
8	robbody
9	robot
10	cleg
11	cseat
12	chair
13	trashcan

Table 9-2: Image Correspondences for Data Tables

VERTICAL INDEX	SURFACE CLUSTER REGIONS
1	20,21,30
2	27
3	16,26
4	8
5	29
6	33,34,35,36,37
7	12,18,31
8	9,28,38
9	17,19,22,25,32
10	20,21,27,30
11	8,16,26,29
12	9,12,18,28,31,38
13	9,12,17,18,19,22,25,28,31,32,38
14	8,16,17,19,22,25,26,29,32
15	8,9,12,16,17,18,19,22,25,26,28,29,31,32,38
16	8,16,20,21,26,27,29,30
17	8,16,17,19,20,21,22,25,26,27,29,30,32
18	8,9,12,16,17,18,19,20,21,22,25,26,27,28,29,30,31,32,38

Table 9–3: Final Plausibilities for Each Surface Cluster

SC	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-36	-29	-25	-27	-36	-29	-32	-30	-30	-30	-45	-33	-39	-43
2	-50	-42	-42	-48	-61	24	-38	-49	-30	-45	-41	-63	-55	-53
3	-86	-39	-55	-58	44	-25	-13	-46	-26	-47	-75	-77	-77	-43
4	-78	-46	-52	-51	-53	-39	-48	-51	7	-33	-49	-58	-45	-47
5	-76	-67	-66	-73	-72	26	-46	-66	-47	-63	-66	-86	-78	-67
6	-39	-28	-34	-31	-46	-49	-48	-39	-60	-50	-19	-29	-34	-71
7	-52	22	-26	-15	-27	-54	-46	-36	-51	-49	-15	-42	-32	-49
8	-68	-41	-48	-53	-38	-34	-45	-57	17	-40	-53	-80	-54	33
9	-11	0	3	1	-34	-34	-35	-23	-52	-38	-51	-58	-45	-44
10	-41	-34	-30	-32	-38	20	-21	-29	-29	-34	-41	-38	-43	-48
11	-78	-35	-48	-49	33	17	27	-32	-1	-27	-56	-64	-51	-41
12	-54	15	-27	-17	-29	-33	-38	-34	18	-24	-17	-44	-29	29
13	-18	18	-3	5	-27	-31	-36	-28	18	-17	-17	-44	-29	30
14	-20	-8	-5	-6	36	17	26	5	-1	-2	-56	-65	-52	-41
15	-20	16	-4	4	35	17	26	7	16	3	-18	-46	-30	29
16	-45	-33	-32	-36	36	17	26	-18	-1	-18	-44	-42	-41	-41
17	-19	-7	-5	-6	35	17	27	5	-1	-2	-44	-41	-40	-41
18	-20	16	-4	4	35	17	26	7	16	3	-18	-42	-27	29

Table 9–4: Supercomponent Evidence Plausibilities

SC	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-29	-27	-27	-29	-21	-21	-29	-30	-30	-99	-39	-39	-99	-99
2	-34	-32	-32	-29	-21	-21	-29	-34	-34	-99	-43	-43	-99	-99
3	-35	-49	-49	-32	27	27	-32	-27	-27	-99	-51	-51	-99	-99
4	-35	-49	-49	-32	27	27	-32	-27	-27	-99	-45	-45	-99	-99
5	-35	-49	-49	-32	27	27	-32	-27	-27	-99	-51	-51	-99	-99
6	-28	-31	-31	-39	-48	-48	-39	-50	-50	-99	-34	-34	-99	-99
7	22	-15	-15	-34	-38	-38	-34	-24	-24	-99	-29	-29	-99	-99
8	15	-17	-17	-34	-38	-38	-34	-24	-24	-99	-29	-29	-99	-99
9	18	5	5	5	26	26	5	-2	-2	-99	-29	-29	-99	-99
10	-33	-32	-32	-18	26	26	-18	-18	-18	-99	-41	-41	-99	-99
11	-8	-6	-6	5	27	27	5	-2	-2	-99	-41	-41	-99	-99
12	18	5	5	-28	-36	-36	-28	-17	-17	-99	-29	-29	-99	-99
13	18	5	5	7	26	26	7	3	3	-99	-29	-29	-99	-99
14	16	4	4	7	27	27	7	3	3	-99	-30	-30	-99	-99
15	16	4	4	7	26	26	7	3	3	-99	-27	-27	-99	-99
16	-7	-6	-6	5	27	27	5	-2	-2	-99	-40	-40	-99	-99
17	16	4	4	7	27	27	7	3	3	-99	-27	-27	-99	-99
18	16	4	4	7	26	26	7	3	3	-99	-27	-27	-99	-99

The tables are a bit overwhelming, but one thing they illustrate is the diversity of plausibility values, arising from various sources. As an example, the trashcan ASSEMBLY is model 13 appearing in surface cluster 8 (among others). It has subcomponent evidence 0.37, and inhibition of 0.17, resulting in an integrated value of 0.33. These values are circled above.

To simplify presentation, all ASSEMBLY invocations for this image are summarized in table 9–8. As commented in section 9.3, successful invocations in one context mask invocations in larger containing contexts. Hence, not all positive final plausibilities from table 9–3 cause invocation.

Table 9–5: Subcomponent Evidence Plausibilities

SC	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-36	-30	-27	-27	-36	-29	-32	-30	-30	-30	-45	-33	-39	-43
2	-44	-37	-38	-42	-55	24	-32	-43	-24	-40	-35	-57	-49	-47
3	-75	-28	-46	-47	41	-17	-2	-35	-15	-36	-64	-66	-66	-32
4	-77	-44	-53	-49	-54	-40	-46	-49	7	-31	-47	-56	-43	-45
5	-70	-61	-61	-66	-69	23	-39	-59	-41	-57	-60	-80	-72	-60
6	-39	-28	-36	-31	-46	-49	-48	-39	-60	-50	-19	-29	-34	-71
7	-48	22	-22	-9	-22	-49	-41	-31	-45	-44	-9	-37	-26	-43
8	-61	-33	-42	-44	-29	-26	-36	-49	25	-32	-44	-72	-46	37
9	-12	0	0	1	-36	-36	-34	-22	-51	-37	-50	-57	-44	-43
10	-36	-29	-27	-27	-36	24	-16	-24	-24	-29	-35	-33	-38	-43
11	-70	-27	-42	-42	41	23	35	-24	7	-19	-47	-56	-43	-32
12	-48	22	-22	-9	-22	-26	-30	-26	25	-16	-9	-37	-21	37
13	-12	25	1	12	-22	-26	-29	-21	25	-9	-9	-37	-21	37
14	-12	0	0	1	41	23	35	14	7	6	-47	-56	-43	-32
15	-12	25	1	12	42	23	35	16	25	13	-9	-37	-21	37
16	-36	-25	-26	-27	41	24	35	-9	7	-9	-35	-33	-32	-32
17	-12	0	0	1	41	24	35	14	7	6	-35	-33	-32	-32
18	-12	25	1	12	42	24	35	16	25	13	-9	-33	-18	37

Invocation was selective with 17 invocations of a possible 252 and many of the invocations were correct (10 of 17). Of the correct, 9 were in the smallest correct context and 1 was in a larger context. All appropriate invocations occurred. Of the incorrect, only 3 were unjustified (notes 2 and 3 in table 9-8).

Though the surface model invocations are not shown, 24 invocation were made out of 475 possible. Of these, 10 were correct, 10 were justifiably incorrect because of similarity and 4 were inappropriate invocations.

All unjustifiably invoked models were eliminated during hypothesis completion and verification (chapters 10 and 11).

Clearly, for this image, the invocation process works well. For the second image, the results were:

- surfaces – 7 correct invocations, 1 justifiably incorrect and 1 incorrect out of 300 possible, and
- assemblies – 9 correct and 1 incorrect of 266 possible.

The chief causes for improper invocation were:

1. not large enough context to contain all subcomponents coupled with having surfaces not contained in the successful context (resulting in a context not directly related and hence not suppressed by the correct context), and
2. superficial similarity between features.

Possible solutions to these are:

- improving the depth merged surface cluster formation process to give better contexts and
- increasing the number and discrimination of the direct evidence.

The preference weights used for the subcomponent and direct evidence calculation were almost always the same. Hence, all evidence was equivalent for

Table 9–6: Association Evidence Plausibilities

SC	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
2	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
3	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
4	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
5	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
6	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
7	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
8	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
9	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
10	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
11	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
12	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
13	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
14	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
15	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
16	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
17	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
18	-99	3	22	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99

Table 9–7: Inhibition Plausibilities

SC	0	1	2	3	4	5	6	7	8	9	10	11	12	13
1	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
2	24	24	24	24	24	-99	24	24	24	24	24	24	24	24
3	44	44	44	44	-99	44	44	44	44	44	44	44	44	44
4	7	7	7	7	7	7	7	7	-99	7	7	7	7	7
5	26	26	26	26	26	-99	26	26	26	26	26	26	26	26
6	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99	-99
7	22	-99	22	22	22	22	22	22	22	22	22	22	22	22
8	33	33	33	33	33	33	33	33	33	33	33	33	33	(17)
9	3	3	1	3	3	3	3	3	3	3	3	3	3	3
10	20	20	20	20	20	24	20	20	20	20	20	20	20	20
11	33	33	33	33	44	33	33	33	33	33	33	33	33	33
12	29	29	29	29	29	29	29	29	29	29	29	29	29	33
13	30	30	30	30	30	30	30	30	30	30	30	30	30	29
14	36	36	36	36	33	36	36	36	36	36	36	36	36	36
15	35	35	35	35	36	35	35	35	35	35	35	35	35	35
16	36	36	36	36	33	36	36	36	36	36	36	36	36	36
17	35	35	35	35	36	35	35	35	35	35	35	35	35	35
18	35	35	35	35	35	35	35	35	35	35	35	35	35	35

these calculations. Higher performance might have been achieved by a judicious selection of weight values, but this was not investigated. Since performance was good even though they were not used, they might be eliminated. This suggests invocation is more affected by the topology of the network and quantity of evidence than by the relative importance of different types of evidence. However, in situations where many concepts are linked with varying strengths of association, the weighted version may be more appropriate.

Table 9–8: Invoked Hypotheses for Image 1

MODEL	SURFACE CLUSTER	PLAUSIBILITY	INVOCATION STATUS	NOTES
robshldbd	3	0.45	E	
trashcan	8	0.34	E	
robshould	10	0.28	E	
robshldsobj	5	0.26	E	
robshldsobj	2	0.24	I	3
lowerarm	7	0.23	E	
robshldsobj	10	0.20	I	3
robbody	13	0.18	I	1
robbody	12	0.18	I	1
robbody	15	0.17	L	
robbody	8	0.17	I	1
link	15	0.08	E	
link	17	0.06	I	4
upperasm	13	0.05	E	
robot	15	0.04	E	
upperarm	9	0.03	E	
lowerarm	9	0.00	I	2

STATUS

E – invocation in exact context

L – invocation in larger context than necessary

I – invalid invocation

NOTES

1 – because trashcan outer surface very similar

2 – similarity with upperarm model

3 – ASSEMBLY with only surface has poor discrimination

4 – not large enough context to contain all components

Extensions To Invocation

Several extensions to the implemented invocation process are proposed:

- boundary symbols: Configurations of 2D and 3D boundary segments also form symbolic objects, as in the heart figure (9-7) or as in cartoons or sketches. Boundary structures could also acquire low-level object independent labels, such as "straight line", "right angle", or "curvature discontinuity point".

- more description types: More types would increase the relative discriminative power of direct evidence. The descriptions could be for curves, surfaces or solids.

- spatial configurations: The process leading to an invocation from a configuration, as in figure 9-6, has not been defined.

- object independent low-level symbols: No low-level, identity-independent symbols were implemented. Adding these should increase the discrimination and descriptiveness of the conceptual network. The richer descriptions would cause faster recognition through invocation instead of through explicit hypothesis completion. The key difficulty with low-level vocabulary is not of invocation, but of verification without explicit models.

- uniform treatment of direct and subcomponent evidence: If both direct evidence and subcomponents are generalized to object features, then the two evidence computations can be unified. This is particularly necessary when low-level vocabulary eliminates direct access to image features.

- identified subcomponent groupings: If the typical subcomponent visibility groups were made into explicit symbols (i.e. identities), then much of the complication of the subcomponent evidence calculation could be simplified, especially if subcomponents are generalized to subfeatures.

- quantised descriptor units: Marr ([MAR82]) proposed that descriptions should be symbolic rather than numerical. Hence, direct evidence could be encoded into symbols representing value ranges (e.g. "much larger", "larger", "equivalent", "smaller" and "much smaller"). This would require a new evaluation function to compare descriptions and might reduce sensitivity.

- generic exceptions: Occasionally properties of the supertypes are not held by the subtype. For example, the prototype ripe apple is red, whereas there are many green and yellow ripe apples. So, there needs to be a way to formally override constraints.

Formation of The Network

There are three open problems that this work raises:

1. How is the structure of the model network created and modified?

2. How are the features used for invocation selected?

3. How are the evidence constraint and association weight values chosen?

These are all "learning" questions that address the problem of creating the network structure used for invocation. These questions were not investigated.

Criticisms of the Invocation Theory

One deficiency is the absence of a justified formal criterion for when to invoke a model. Currently, if the combined plausibility measure (section 9.2.8) is positive, then the model is invoked. This has worked well in practice, leading to few false invocations. Seriously incorrect hypotheses are often near -1.0, so a threshold somewhat lower than 0.0 could be considered. This might lead to each object needing a different threshold. On the other hand, the positive minus twice

negative average and the inhibition formulation distinguish the zero level as a key threshold between supportive and contradictory evidence, so this suggests that the 0.0 level should be kept.

Another difficulty with the theory is deciding when to stop invoking. Recognition of one structure strongly boosts the plausibilities of its supercomponents in contexts that may not contain the supercomponent. This could lead to spurious invocations.

This thesis has used the term "generic" relationships somewhat loosely and has not distinguished between whether such a relationship between A and B (supertype of A is B) meant a member of class A was also a member of class B, or A was a member of class B, or whether there was merely some conceptual relationship based on complexity and detail between the two. The precise definition is not important to the network and its computation, and what it can be taken to mean is that anything callable by A could also be callable by B.

The final major criticism is over duplication of invocation. Because of the type structure, any type invocation should lead to a supertype invocation. There may also be multiple class invocations (e.g. < *wheel* > and < *automobile part* > in figure 9–15). Finally, because of the nesting of surface cluster contexts, there is the possibility of invoking the same object in several contexts. The net result is that one real object may cause, for example, 3 types * 2 classes * 3 contexts = 18 separate invocations, all with slightly different plausibilities. There doesn't seem to be any ideal way to control this. The multiple type problem could be solved by invoking at the most general type, verifying there, and then letting the subtype constraints refine the result to the most specific. The multiple class problem must be accepted as intrinsic, leading to multiple invocations. This is necessary because general visual goals are not sufficient to determine which solution is the best. The context problem is partly solved by having identities in subcontexts inhibit those in containing contexts. Verification also disables invocation in larger and smaller contexts, as the object is already found.

Some minor criticisms arise because representational scale, boundary description and non-shape properties have been largely ignored in the analysis.

Invocation Contributions

The research presented in this chapter makes the following original contributions:

- The formalization of the associative basis for the invocation process with the major elements as object types, direct evidence inputs and associative links based on generic and component relations. This also includes formalizing invocation as a plausibility calculation in an association network.

- The elucidation of constraints on how different evidence affects plausibility, and the implementation of these constraints as algorithms.

- Demonstration of surfaces and surface clusters as the contexts in which to consider invocation.

- Proposing a model for how such a network could be implemented on a 2D or 3D parallel machine in registration with image data.

- Demonstration of a successful implementation of the theory, albeit on only a few examples.

Chapter 10

Hypothesis Completion

At this point the recognition process has isolated a set of data, described its features and invoked a model as its potential identity. To say the object is genuinely recognized requires having a pairing between features in the model and data from the image. An important concurrent activity is determining where the object is relative to the viewer. The hypothesis completion process has the goal of fully instantiating correctly invoked models, estimating object 3D position and accounting for occlusion, whether by the object itself or by external objects. Section 1 discusses intuitions behind the theories, which are developed in section 2. Section 3 presents critical discussion on the topic.

10.1 Intuitions on Finding Features

Without direct correspondences between model features and image evidence, object recognition is only suggestive. It is like saying that a collection of gears and springs is a watch. Further, for 3D scene understanding, 3D correspondences must be established, so object position must be found. Not knowing the object's location also presents a practical problem as it is required for most uses of object identification (e.g. robot assembly). Hence, locating and orienting the object and making image-data correspondences are necessary parts of a general vision system. These tasks are the first part of substantiating the existence and identity of the object, the final stages of which are done in verification (chapter 11).

Why Collecting All Evidence is Desirable

The hypothesis construction process should find as much evidence as possible. Ideally, it would find direct image evidence for all model features, but this is impossible. Resolution changes might make the information too large or small to directly detect, and occlusion will hide some of it, assuming a single point of view (which is the case here).

Why should maximal evidence be collected? An ideal domain would have all objects extremely distinguishable using only a few attributes, and this is largely true for the broad classes we traditionally consider as distinguished, like trees, cars and people. However, to identify subclasses or individuals requires finer details (e.g. distinguishing between two people, or two "identical" twins).

Many details are object-specific, and a goal-directed argument suggests that only the key differentiating feature need be found. When the domain is sufficiently restricted, specific features will be unique signifiers; alternatively, two objects could be discriminable using a few features. However, this would be an inappropriate strategy for a general vision system because, without additional descriptions or external, non-visual knowledge of the restricted domain, it would not ordinarily be possible to reach the stage where only a few identities were under consideration. Although simple model bases admit a decision-tree type solution, this is not the best general approach.

Many individual objects differ only slightly or share identical features. Consider how often one recognizes a facial feature or a smile of a friend in the face of a complete stranger. Though the stranger is unique through the configuration of her features, some details are held in common with the friend. If recognition were predicated on only a few features, which may sometimes be sufficient for unique identification in a limited domain, then we would be continually mis-recognizing objects. While only a few may be necessary for model invocation, many others are necessary for confirmation.

Partial evidence is often sufficient. We usually have no trouble identifying a friend even when a mustache has been shaved off, and often do not even notice

that there is a difference, let alone know what the difference is. Yet, we tend to expect recognition to be perfect. So, on idealistic grounds, a general vision system should should acquire as much information as possible.

The level of detail in a model affects the quantity of evidence required. Hierarchical models that represent finer details in lower levels of the model lead to hypothesis completion processes that add the details once the coarser description is satisfied (if the details are needed). Hence, some evidence might not be needed at a particular level. This approach was not pursued.

In summary, full model instantiation derives from:

- a philosophical requirement – that true image understanding requires consistent interpretation of all visible features relative to a model and contingent explanation of missing features,

- a practical requirement – that many details are needed to distinguish similar objects, especially as many objects share common features and some details will be absent for environmental reasons (e.g. occlusion), and

- a modeling requirement – that objects should be recognized to the degree they need to be distinguished.

What Counts as Evidence

What is desired is image evidence that supports the existence of each model feature. Ideally, there should be a direct correspondence. In edge-based recognition systems, an image edge was the ideal candidate, because surface orientation discontinuity boundaries appeared as edges and these could be easily paired. This was even more important in polyhedral domains (without reflectance boundaries), where extremal boundaries were also orientation discontinuity boundaries, and so made pairing easier. Unfortunately, more naturally shaped and colored objects led to a veritable plethora of new problems: there were fewer traditional orientation edges, extremal boundaries no longer corresponded to

orientation boundaries and reflectance variations created new edges. So, these made the search for simple and directly corresponding image boundary evidence much more difficult.

Two of the advantages of using surfaces given in chapter 3, are mentioned here again:

- using surfaces as the primary representational unit of both the raw data and the object model makes the transformation distance between the two almost non-existent, and

- the interpretation of a surface data unit is unambiguous (unlike image edges which may correspond to a variety of scene phenomena).

With surface representations, it is again possible to find image evidence that directly associates with model features. Under the bold assumption that there is a consistent description regimen (e.g. segmentation) for both the surface image and model surfaces, the model feature instantiation problem can be reduced to finding which model surface element corresponds with each data surface element. This assumes that surfaces are the primitive model feature, which is the case here. Hence, surface data should considerably advance the goal of robust recognition.

One result of using the surface segmentation proposed in chapter 3 is a discrete symbolic partitioning of the complete object surface. This simplifies the surface matching computation tremendously. An infinitesimal element of a surface could have many possible identities and this shows up in practice as the need to incrementally rotate and shift surfaces when doing matching (e.g. [IKE81] or reduced by scale [POT83]). A segmented surface immediately simplifies the matching by choosing a higher level structure for comparison. Topology further decreases the amount of matching as adjacent model surfaces must pair with adjacent data surfaces, reducing the problem to subgraph isomorphism. And, if the invocation process gives strong suggestions to the identity of the various surfaces, then combinatorial matching is almost completely unnecessary.

Unfortunately, this matching is based on a canonical segmentation of the surface data, which is clearly unlikely. Scale changes affect the shape, position and number of surfaces produced by segmentation. Further, surfaces will be partially out of view due to object rotation, and this also creates unexpected segments. Both of these make the matching process more difficult.

Chapter 3 proposed criteria for canonical segmentation of fully visible regions at a single scale. As segmentation and comparison across multiple levels of scale is not considered here, we can assume that visible image surfaces will be directly matchable to model surfaces. The self-occlusion and observer viewpoint problems are explored below.

Up to this point, surfaces are the only evidence accepted for model features. The other type of evidence is pre-assembled collections of surfaces represented as distinct, nameable sub-objects of the object hypothesis currently being constructed. For example, an nose would be such a subassembly in the context of a face. The structures can be rigidly connected to the parent assembly (e.g. nose to face) or flexibly connected (e.g. arm to body). The collections should correspond to model features because of the model segmentation assumptions (chapter 5) and the surface cluster formation process (chapter 7).

Any analysis associated with these structures can be reduced to analysis of the subcomponent surfaces, but it would be desirable to use the larger units. First, the substructures might have been previously identified, and so processing should not be duplicated, and second, the use of larger conceptual units reduces the computational load arising from combinatorial matching. Finally, parsimony dictates that matching should proceed at the level of descriptions, and complex objects would be described using sub-objects.

Rigid substructures have similar requirements as surfaces: their positions must be reconciled with the main object's position, and adjacent structures meet standard surface adjacency criteria. Flexibly connected substructures have different constraints: their reference frames will have one or more relative degrees of freedom, but the other location parameters are constrained by definition. In ei-

ther case, the substructure identity is specified, which tightly constrains possible matches.

Determining Location and Orientation

Part of competent object recognition is knowing where an object is – hence its 3D location and orientation must be determined. This information is also needed internally, as identity verification requires finding all visible object features correctly placed. Knowing the object's spatial reference frame enables prediction of image locations for the features.

Invocation suggests a few data-to-model feature (e.g. surface) correspondences to form an initial "island of stability". From this, the reference frame of the object relative to the viewer can be deduced by comparing the geometrical relationships between the image data with those implicit in the model.

A single surface usually provides only identity and surface normal data. Assuming correct correspondence with the model surface, it constrains the object to a single rotational degree of freedom about the surface normal. A second rotation axis, whether from a second surface or from an axis of curvature on the first surface, usually completely constrains the object's orientation (possibly up to a mirror image).

Individual surfaces may also be assigned a reference frame directly. Because the object models may include boundary descriptions, the final degree of rotational freedom can be estimated by rotating the model surface until it achieves the best boundary correspondence. This requires considering surface shape discontinuity boundaries as candidates for matching. The back-side-obscuring boundaries are clearly caused by unrelated structure and only the front-side-obscuring boundaries that occur at surface orientation discontinuities have direct model correspondences (i.e. not at tangential boundaries).

Because the data is seldom perfect, it is assumed that multiple features will generate different location and orientation estimates. Hence, reconciliation will be necessary, which should also reduce data errors.

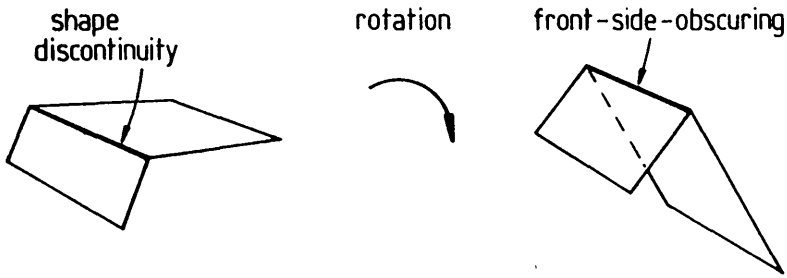


Figure 10–1: Boundary Type Changes During Surface Rotation

How to Find Features

Finding all model features requires understanding how 3D objects appear in images – to locate image evidence for oriented model instances. The features represented in this thesis are surface patches with particular boundary segmentations, so the recognition process must understand or at least be able to predict how patch appearance varies with changes in the surface's position relative to the viewer. The segmentation process attempts to produce surface patches with a uniform curvature characterization, so it is easy to approximate the visible shape to first order, given the model patch and its relative position.

The hypothesis completion process should also understand how boundary appearances change as a function of viewpoint. In particular, shape discontinuity boundaries disappear when the viewpoint becomes tangential to the surface and are replaced by new front-side-obscuring type boundaries (i.e. the surface starts to obscure itself). Figure 10–1 illustrates this boundary transformation. Boundary appearance understanding was not pursued.

Occlusion

Because distant objects can be partially obscured by closer objects indirect evidence for some features must be found. This comes in two forms – evidence for closer structures and validation that the available features up to the point of occlusion are consistent with the model.

There are several distinct cases of occlusion. The first case is degenerate: there are features on the back side of every object and these cannot ordinarily be detected from a single viewpoint (except by using mirrors or shadows). At the same time, it is easy to predict what cannot be seen, using the estimated orientation of hypotheses to predict back-facing surfaces.

The next occlusion case is forward-facing self-occlusion, whether partial or complete. Here, an object feature (e.g. surface) is obscured by one or more closer surfaces from the same object. Given knowledge of the object's position relative to the viewer, the relative surface positions and their visibility can be predicted.

Finally, there is the case of front-facing structure obscured by unrelated objects. Because the obscuring objects are unrelated, the details of occlusion cannot be predicted, nor is it be possible to deduce the invisible structure (though context and historical information could help – as in the top of a desk). Perhaps the best that can be done is to show that what remains is consistent with the hypothesis of obscured structure. Knowing the obscuring objects and their position can help predict where occlusion will take place, but gives no information to the obscured structure of unrelated objects (unlike related objects in the self-obscured case).

Occlusion is not the only phenomenon that causes data to be missing. In particular, there are faulty objects, sensor noise, generic object variations, scale related segmentation variations and non-scale segmentation variations. These other problems are not considered here.

In conclusion, hypothesis completion must accomplish three tasks:

- discover the object's local reference frame,

- find as much evidence as possible for model features, and
- properly explain all missing data as instances of occlusion.

10.2 Techniques for Hypothesis Completion

This section presents techniques for solving three classes of problems:

1. explicitly estimating the reference frame for structures,
2. predicting features not visible because of self-occlusion, and
3. finding suitable evidence for the presence or absence of other model features.

The ordering of material follows the sequence of events in the hypothesis construction process. The process starts with a hypothesis from invocation, which will have several model-data correspondences assigned. From these correspondences, an initial estimate of the structure's position and orientation is calculated. This estimate is used for predicting which object features are invisible because of being back-facing or being obscured by closer object surfaces. Finally, the process acquires valid evidence for each of the remaining model features or tries to use occlusion by unrelated objects to explain their absence.

10.2.1 Reference Frame Estimation

Reference Frame Representation

Before discussing the estimation of reference frames for surfaces and solids, some comment is needed on how the individual parameter estimates are represented and how estimates from separate information sources are integrated.

Two contenders for reference frame representation were ACRONYM's method ([BRO81]) and Faugeras and Hebert's method ([FAU83]). ACRONYM's advantage was that it could easily integrate new evidence by adding new constraints. Its disadvantage was that the current estimate for a parameter was implicit and could only be obtained explicitly by substantial algebraic constraint manipulation, which result only a range of values with no measure of "best". In any case, such a mechanism was not available here. Faugeras and Hebert used a least-squares method to estimate a "best" rotation and translation estimate from a set of correspondences. New data just meant a larger set over which the best estimate was estimated. Its advantage was it integrated the set of data uniformly to give a good estimate, but at some computational cost. Further, its criteria for inconsistency is based on the accumulated error. In retrospect, probably either of these methods would have been an improvement on the method actually used, which is now presented to set the context for the major results of this chapter.

Each individual parameter estimate is expected to have some error, so it is represented by a range of values. (The size of the range was acquired by experience.) Thus, an object's position is represented by a 6 dimensional parameter volume, within which the true parameter vector should lie.

Integrating parameter estimates is by intersecting the individual parameter volumes. All the 6D parameter volumes are "rectangular solids" with all "faces" parallel, so the intersection is easily calculated and results in a similar solid. By the assumption that the true value is contained in each individual volume, it must also lie in the intersection. The effect of multiple estimates is to refine the tolerance zone by progressively intersecting off portions of the parameter volume, while still tolerating errors.

If a final single estimate is needed, the average of each pair of limits is used. An example of the use of this method is given below, when estimating the reference frame of an ASSEMBLY.

Up to now, the transformation of coordinate reference systems has been by multiplication of the homogeneous coordinate matrices representing the transforms. Since we are now using a parameter estimate range, the transformation

computation must be modified. In the most general case, each transformation would have its own range (to allow for object flexibility), but, as implemented here, only the map from the camera coordinate system to the object is allowed variations. These variations propagate through the calculation of the global or image locations for any feature specified in any level of reference frame. The variation affects two calculations:

1. how to calculate a combined transformation given that one transformation is a range, and

2. how to calculate the range of positions for a point given a transformation range.

The technique used for both of these problems is similar, and is only an approximation to a correct solution:

1. For a subset of values in the parameter range,

   (a) Calculate a transformation

   (b) Transform the second parameter vector (or point)

2. Bound the set of mapped vectors (or points)

This process is illustrated in figure 10–2. In (a), a 2D parameter range with the subset of points is designated. In (b), the original range is rotated to a new range, and (c) shows the parameter bounds for the new range.

The figure illustrates one problem with the method – parameter bounds are aligned parallel with the coordinate axes, so the parameter area (6D volume) increases with each mapping. A second problem is that the rotation parameter space is not rigid in this coordinate system, so the shape of the parameter space can change greatly. If it expands in a direction not parallel with a coordinate axis, the combination of the first problem with this one can result in a greatly expanded parameter space. Further, the mapping is not unique, as zero slant

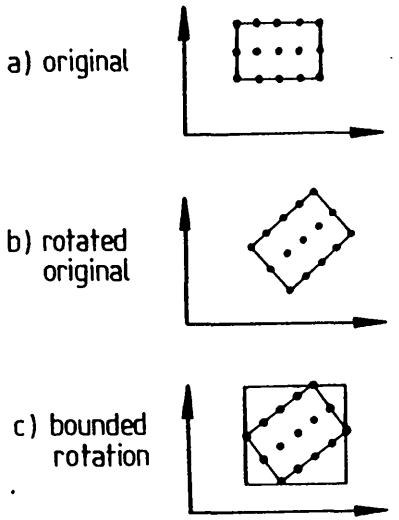


Figure 10–2: 2D Rotation of Parameter Ranges

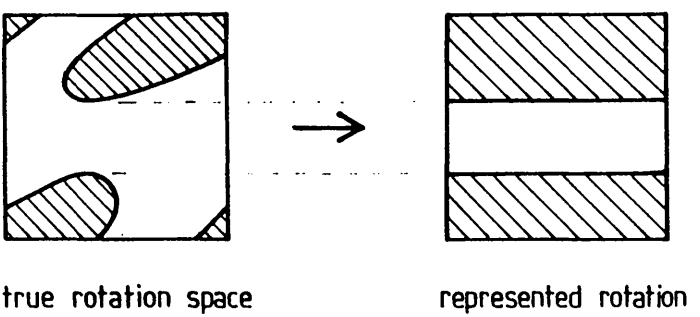


Figure 10–3: A Difficult Parameter Space

allows any tilt, so any transformations that include this can grow quickly. Figure 10-3 shows a difficult 2D parameter region and its bounding representation.

There is also a programming problem because the angular parameter space is "circular", and mapping parameters may cause a wrap around. This causes a difficulty in determining the interior of the parameter volume after rotations, if only sparse points are used in the mapping heuristic used above. The solution adopted was to use 5 points for each angular parameter (total 5^3 points) and bound the resulting set of points. This was time-consuming.

One general problem with the method is that consistent data does not vary the parameter bounds much, so that intersecting several estimates does not improve the bounding greatly. Hence, there is still a problem with getting a "best" estimate from the range. Another problem with the general case is each model variable increases the dimensionality of the parameter space, requiring increased computation and compounding bounding problems.

The conclusion is that this method of representing and manipulating parameter estimates is not adequate.

Estimating Individual Surface Reference Frames

Individual surfaces have local reference frames attached when invocation suggests a potential identity. The surface's spatial position is represented by the transformation from the camera coordinate frame to this local one. The key problem is to deduce the transformation.

Several constraints are available to make this possible. Fisher ([FIS83]) showed how the transformation could be deduced using the boundary shape. Estimation of the orientation parameters (rotation, slant and tilt) used the cross-section width as a function of image angle, which deforms in a characterizable way. In the research presented here, surface orientation is directly available, so only one rotational degree of freedom needs to be resolved. Figure 10-4 shows a planar data patch and a hypothesized model patch with surface normals. For

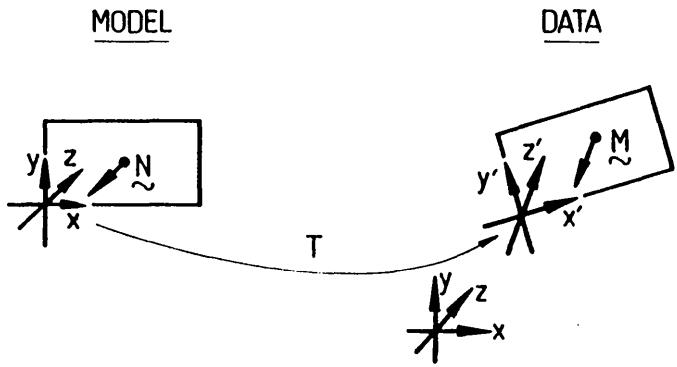


Figure 10–4: Transformation Linking Model to Data Surface

these to be the same, the normal vectors must be parallel. Hence, in 3D the model surface can rotate only about the normal.

The final rotation is estimated by correlating the angular cross-section width as a function of rotation angle. Figure 10–5 illustrates this. For non-planar surfaces, an approximate solution is obtained by using the normals at the centroid of the surfaces. A more complete solution using the curvature orientation is presented below.

Though occlusion may prevent the whole surface from being visible, data cross-sections ending on a back-side occluding boundary are still included, because they contribute some evidence. Further, surface reconstruction (chapter 6) may eliminate some of the problem with data loss.

The complete method is:

Rotate image surface until the central point normal is
aligned with the -Z camera axis (R_1)

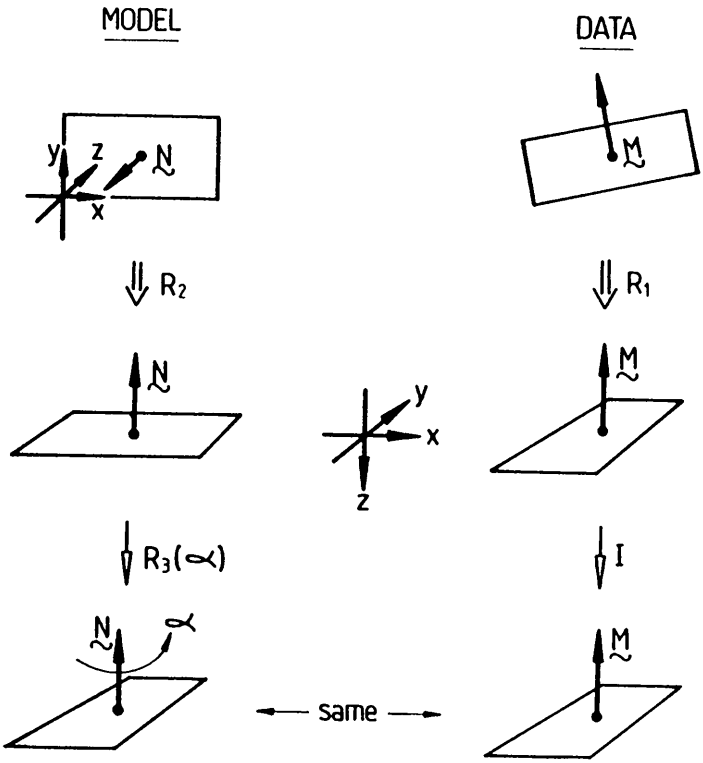


Figure 10–5: Estimation of Rotation for Isolated Surface Patches

Rotate the model surface until the central point normal is
aligned with the -Z camera axis (R_2)
Calculate data surface cross-section widths
Calculate model surface cross-section widths
For each rotation angle (α) about the model normal axis:
calculate model rotation ($R_3(\alpha)$)
correlate cross-section widths
Set a threshold = 0.9 * peak correlation
Pick peak correlations (α_i)
(If more than 30% above threshold, declare circularly
symmetric: $\alpha_i = 0.0$)
Solve for reference frames: $R_1^{-1} R_3(\alpha_i) R_2$
Get algebraic solution for three rotation angles for each
reference frame

Given the rotations, the translations are estimated. Fisher ([FIS83]) estimated these directly from the boundary data. Depth was estimated by comparing model to data areas and cross-section widths. The 3D translation was estimated using the 2D translation that best fitted the data and then inverting the projection relationship using the estimated depth. This research has depth estimates directly available, and the x,y translation is estimated by relating the model surface centroid to the 2D image centroid and inverting the projection relationship.

The estimated and nominal translation and rotation values for the modeled surfaces successfully invoked in the test images are given in tables 10–1 and 10–2. All values here are in the camera reference frame.

The rotation estimates are good, even on small surfaces (robshoulds) or partially obscured surfaces (uside, uends, lsideb, ledgea). The translation estimates are also reasonable, but not as accurate. Surfaces lsideb, ledgea and uside were substantially obscured, yet their translation estimates are reasonable. Using the boundary instead of just a central point should reduce their error. For the un-

Table 10–1: Translation Parameters for Single Surfaces

Surface Name	Test Image	Image Region	Measured (cm)			Estimated (cm)		
			X	Y	Z	X	Y	Z
robshldend	1	26	-31.5	22.8	556	-22.0	18.6	558
robbodyside	1	8	-13.9	-32.4	565	-13.5	-35.9	562
robshoulds	1	29	-20.9	17.8	556	-16.2	9.8	564
uside	1	19,22	-21.0	13.4	585	-13.6	31.0	578
uends	1	25	27.2	16.6	547	35.6	16.5	551
lsideb	1	12	23.7	16.9	533	21.9	28.4	539
ledgea	1	18	23.7	16.9	533	26.6	16.4	538
tcanoutf	1	9	22.3	-44.1	536	29.1	-44.2	541
cleg(lf)	2	21	-26.1	18.8	431	-21.2	10.4	423
cleg(lr)	2	24	-24.1	19.9	435	-17.2	11.8	429
cleg(rf)	2	22	15.1	14.6	413	26.3	3.7	422
cleg(rr)	2	23	17.1	15.7	418	23.5	26.6	422
cseat	2	9	-3.5	17.8	427	10.2	9.1	412
cbackf	2	4	-3.5	17.8	427	1.8	9.5	410
tcanoutf	2	7	.04	-28.8	420	3.7	-35.3	409

Table 10–2: Rotation Parameters for Single Surfaces

Surface Name	Test Image	Image Region	Measured (rad)			Estimated (rad)		
			ROT	SLANT	TILT	ROT	SLANT	TILT
robshldend	1	26	*1	0.88	*1	*1	0.88	*1
robbodyside	1	8	0.00	0.13	4.71	0.02	0.13	4.76
robshoulds	1	29	0.05	0.70	6.08	0.02	0.84	6.15
uside	1	19,22	6.04	0.88	3.48	5.20	0.88	4.32
uends	1	25	3.12	0.75	2.75	3.16	0.66	3.21
lsideb	1	12	1.51	0.88	1.73	1.70	0.88	1.54
ledgea	1	18	4.75	0.70	1.38	4.70	0.73	1.44
tcanoutf	1	9	0.00	0.13	4.71	0.02	0.11	4.56
cleg(lf)	2	21	6.18	0.396	3.15	6.09	0.133	1.44
cleg(lr)	2	24	.044	0.783	4.13	0.10	0.540	4.69
cleg(rf)	2	22	6.18	0.396	3.15	6.09	0.133	1.44
cleg(rr)	2	23	.044	0.783	4.13	0.200	0.540	4.59
cseat	2	9	*1	1.33	*1	*1	1.31	*1
cbackf	2	4	6.23	0.469	3.68	0.01	0.436	3.66
tcanoutf	2	7	6.28	0.237	4.71	6.21	0.214	5.26

*1 – rotationally symmetric surface generates unconstrained solution

obscured surfaces, the average translation error for image 1 is (-6.0,1.6,-1.6). An error in estimating the camera coordinate system for test image 1 could explain the average -6.0 cm error in the x translation. In test image 2, the average position error for the unobscured surfaces was (6.5, -8.6, -8.4) which is believed to arise from minor errors in estimating the camera coordinate system. Other sources of error include measurement error (estimated as 1.0 cm and 0.1 radian), image quantization (estimated as 0.6 cm at 5m and 0.002 radian) and errors arising from the approximate nature of the parameter estimations. In any case, the error is about 1.5% of the distance to the object, so the position error is small. Angular estimates are good except for the chair legs, where the small leg widths cause difficulties with estimating surface normals.

Surface orientation can also be estimated without recourse to the boundary if there is significant surface curvature in one direction. Here, the 3D orientation of the major curvature axis constrains the remaining angular degree of freedom to two possible orientations. The mapping from the model normal and curvature axis vectors to those of the data gives the orientation estimate. Data errors complicate the calculation, which is described in detail in the next subsection. Location estimation is as above. Figure 10–6 illustrates the process of rotation estimation using the normal and curvature axis. Table 10–3 lists the results for this case and table 10–4 shows the results obtained by integrating these results with those from table 10–2 (by parameter space intersection).

The curvature based estimation process gives nearly the same results as the boundary based process.

Estimating ASSEMBLY Reference Frames

The next problem considered is how to deduce the transformation from the camera coordinate system to an ASSEMBLY's local reference frame. If a set of model vectors (e.g. surface normals) can be paired with corresponding data vectors, then a least-squares estimate of the transformation could be estimated using methods like that of Faugeras and Hebert ([FAU83]). This integrates all evidence

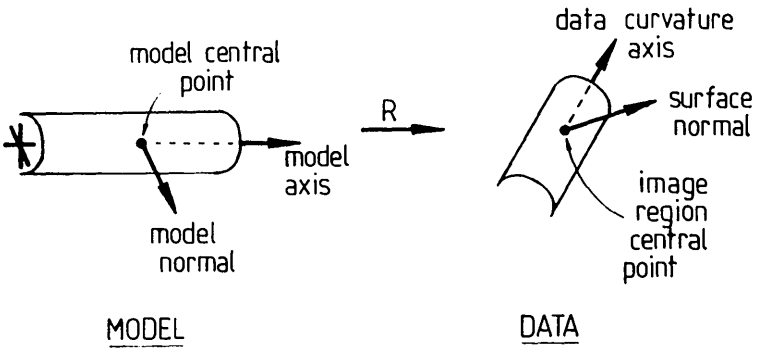


Figure 10–6: Rotation Estimation from Normal and Curvature Axis

Table 10–3: Rotations for Single Surfaces Using Curvature Axis

Surface	Test	Image	Measured (rad)			Estimated (rad)		
Name	Image	Region	ROT	SLANT	TILT	ROT	SLANT	TILT
robbodyside	1	8	0.00	0.13	4.71	6.28	0.13	4.77
robshoulds	1	29	0.05	0.70	6.08	0.10	0.83	6.07
uends	1	25	3.12	0.75	2.75	3.12	0.66	3.24
tcanoutf	1	9	0.00	0.13	4.71	0.01	0.18	4.61
cbackf	2	4	6.23	0.469	3.68	6.14	0.436	3.66
tcanoutf	2	7	6.28	0.237	4.71	6.25	0.271	5.10

Table 10–4: Combined Rotation Parameters for Single Surfaces

Surface Name	Test Image	Image Region	Measured (rad)			Estimated (rad)		
			ROT	SLANT	TILT	ROT	SLANT	TILT
robbodyside	1	8	0.00	0.13	4.71	0.01	0.13	4.76
robshoulds	1	29	0.05	0.70	6.08	0.02	0.83	6.15
uends	1	25	3.12	0.75	2.75	3.16	0.66	3.21
tcanoutf	1	9	0.00	0.13	4.71	0.01	0.18	4.56
cbackf	2	4	6.23	0.469	3.68	6.22	0.436	3.66
tcanoutf	2	7	6.28	0.237	4.71	6.23	0.271	5.26

uniformly. The method described below estimates reference frame parameters from smaller amounts of evidence, which is then integrated using the parameter space intersection method described above. The justification for this approach was that it is incremental and shows the intermediate results more clearly. In retrospect, however, the explicit incremental constraint method of ACRONYM ([BRO81]) would have satisfied these goals better. Better parameter estimation would probably have been achieved using Faugeras and Hebert's method. The success of the method shown below is partly because of the strength of surface evidence.

To start with, each subassembly contributes a position estimate. Suppose, the subassembly has a previously deduced global reference frame G_s and the transformation relationship between the sub-object's reference frame and the main object is A (from the model). (If the subassembly is flexibly connected, then any variables in A will be bound before this step. Section 10.2.3 discusses this.) Then, the estimated new global frame is $G_s A^{-1}$. This case is also used if only one surface is found for an ASSEMBLY. Then, the reference frame of the surface is used for G_s in the above. Figure 10–7 illustrates how the sub-object's reference frame relates to that of the object.

The rest of this subsection considers how position information can be estimated from pairs of surfaces.

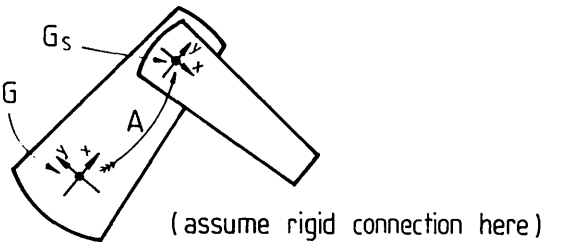


Figure 10–7: Object and Subobject Reference Frame Relationship

Each data surface has a normal that, given correspondence with a particular model surface, constrains the orientation of the ASSEMBLY to a single rotational degree of freedom about the normal. A second, non-parallel, surface normal then fixes the object's rotation. The calculation given here is based on mapping a pair of model surface normals onto a data pair. The model normals have a particular fixed angle between them. Given that the data normals must meet the same constraint, the rotation that maps the model vectors onto the data vectors can be algebraically determined. Figure 10–8 illustrates the relationships.

The above argument used the surface normals as the two vectors, but other possibilities exist. The key observation is that surface normals are reasonable only for nearly planar surfaces. For cylindrical or ellipsoidal surfaces, normals at the central points on the data and model surfaces can be computed and compared, but: (1) small displacements of the measurement point on surfaces with moderate curvature lead to significant changes in orientation, and (2) occlusion makes it impossible to accurately locate central points for data correspondence. Fortunately, cylindrical surfaces have a curvature axis that is more accurately

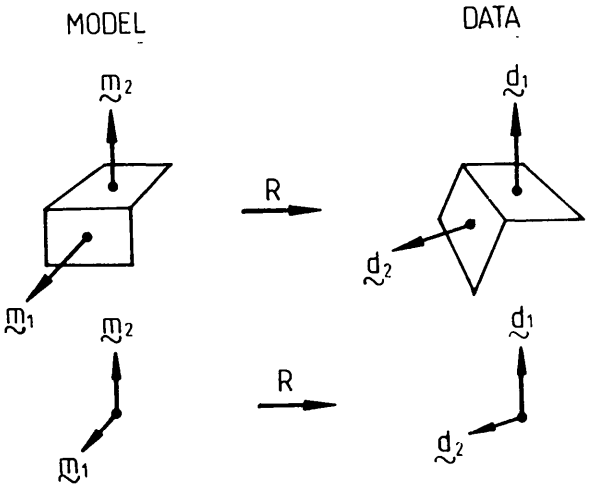


Figure 10–8: Rotating Model Normals to Derive the Reference Frame

estimated and is not dependent on precise point positions nor is affected by occlusion. Figure 10–9 illustrates these points.

Another problem occurs when the two surfaces have nearly parallel vectors (e.g. normals). Here, noise causes wide variations in estimates. This case also has a (proposed) special treatment, using the vector through the central points in the surfaces. This vector is likely to be useful when the points are widely separated. Then, variations in point placement will cause less significant effects in this vector's orientation. This approach is inappropriate when the surfaces are coaxial or are close together. Further, central points are likely to be affected by occlusion. Figure 10–10 illustrates this solution.

Given these observations, eight orientation estimation cases are distinguished:

1. two planes with surface normals not nearly parallel: use the data normals paired to the model normals.

2. two planes with surface normals nearly parallel: use one data normal paired to the model normal and the second vector from paired central points.

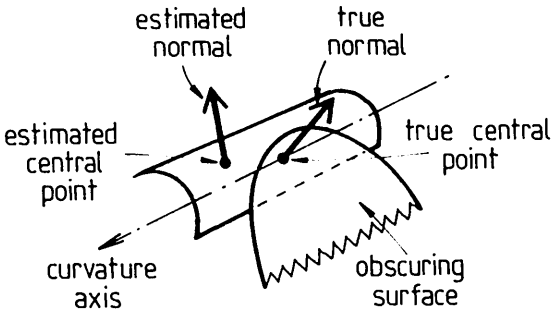


Figure 10–9: Axis Stability on Cylindrical Surfaces

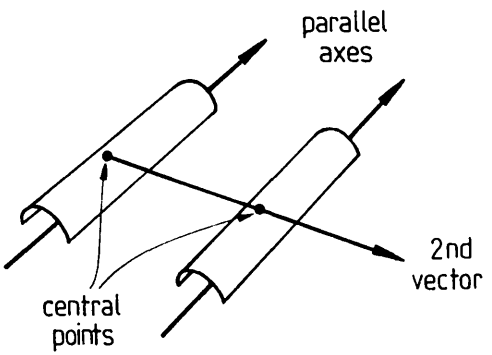


Figure 10–10: Central Points Give a Second Vector

3. anything and an ellipsoid, normals not nearly parallel: use the data normals paired to the model normals.

4. anything and an ellipsoid, normals nearly parallel: use one data normal paired to the model normal and the second vector from paired central points.

5. plane and cylinder, cylinder axis not nearly parallel to plane normal: use paired plane data and model normals, paired cylinder data and model axes.

6. plane and cylinder, cylinder axis nearly parallel to plane normal: use the data normals paired to the model normals.

7. two cylinders, axes not nearly parallel: use data axes paired with model axes

8. two cylinders, axes nearly parallel: use one data axis paired to the model axis and the second vector from paired central points.

Unfortunately, data errors lead to the interior angle between the pairs of vectors being only approximately the same, which makes exact algebraic solution impossible. So, a variation on the rotation method is used. Here, a third pair of vectors, the cross product of each original pair, are calculated and have the property of being at right angles to each of the original pairs.

Let:

$\vec{d_1}$, $\vec{d_2}$ be the data normals

$\vec{m_1}$, $\vec{m_2}$ be the model normals

Then, the cross products are:

$$\vec{c_d} = \vec{d_1} \times \vec{d_2}$$

$$\vec{c_m} = \vec{m_1} \times \vec{m_2}$$

From $\vec{d_1}$ and $\vec{c_d}$ paired to $\vec{m_1}$ and $\vec{c_m}$ an angular parameter estimate can be algebraically calculated. Similarly, $\vec{d_2}$ and $\vec{c_d}$ paired to $\vec{m_2}$ and $\vec{c_m}$ gives another estimate, which is then integrated using the parameter space intersection technique.

Each pair of surfaces contributes two more orientation estimates. (This paired vector process is also used for estimating the orientation of single surfaces with a curvature axis, by using the normal and the curvature axes as the two vectors as described above.)

To illustrate the accuracy of this calculation, two unit vectors (e.g. the model vectors) with a $\pi/3$ separation are mapped to a second pair (e.g. the data vectors) by the transformation: $ROT = \pi/3$, $SLANT = \pi/3$, $TILT = \pi/3$ ($\pi/3 = 1.0471$). Then, the data pair were perturbed so that each vector varies from its true position by a vector with components chosen from a zero mean, 0.05 standard deviation normal distribution. The range of the resulting estimated transformations is:

ROTATION	SLANT	TILT
0.941 – 1.181	0.933 – 1.166	0.888 – 1.186

This shows that the calculation is stable to normal errors.

The global translation estimates come from individual surfaces and substructures. For surfaces, the estimates come from calculating the translation of the nominal central point of the rotated model surface to the central point of the observed surface. Occlusion disturbs this calculation by causing the image central point to not correspond to the projected model point, but the errors introduced by this seemed to be within the general level of error caused by mis-estimating the rotational parameters. Comparing the observed surface boundaries (front-side obscuring and shape) to their positions predicted by the model could improve on the estimates ([FIS83]), but this method was not used here. The implemented algorithm for surfaces was:

Let:

G be the transformation from the ASSEMBLY's coordinate system to that of the camera (i.e. its global position)

A be the transformation from the surface's coordinate system to that of the ASSEMBLY

Then:

1. Get the estimated global rotation for that surface (GA)
2. Rotate the central point ($\vec{p}$) of the model surface ($\vec{V_1} = GA\vec{p}$)
3. Calculate the 3D location of the image region centroid, inverting its image coordinates using the depth value given in the data ($\vec{V_2}$).
4. Estimate the translation as $\vec{V_2} - \vec{V_1}$.

ASSEMBLY Reference Frame Calculation Results

The above theory showed how estimates for the ASSEMBLY's reference frame are calculated. Individual estimates are integrated by the parameter space intersection technique to give a final parameter estimate. The whole process is illustrated by showing the calculation for the robot lower arm.

The rigidly attached hand subassembly is not visible, so it contributes no information. Each of the surfaces paired and transformed according to the above

theory contributed these estimates (in the camera coordinate system):

OBJECT		ROT	SLANT	TILT
lsideb & lendb	MIN	3.966	1.158	4.252
	MAX	0.633	0.204	3.949
lendb & ledgea	MIN	3.487	1.190	4.693
	MAX	0.192	0.216	4.405
ledgea & lsideb	MIN	3.853	1.361	4.599
	MAX	0.430	0.226	4.257

The rotation estimates are integrated by intersection to give the following result:

	ROT	SLANT	TILT
MIN	3.966	1.361	4.693
MAX	0.192	0.204	3.949

and the average value is:

ROT	SLANT	TILT
5.220	2.353	1.180

which compares well with the measured value of:

ROT	SLANT	TILT
5.060	2.236	1.319

Translation is estimated after rotation, and starts with an estimate from each

surface individually. These estimates are:

		X	Y	Z
lsideb	MIN	-1.891	-10.347	503.
	MAX	57.262	48.807	592.
lendb	MIN	-1.206	-26.849	500.
	MAX	58.259	32.616	589.
ledgea	MIN	-1.058	-20.298	503.
	MAX	58.116	38.875	592.

The translation estimates are integrated by intersection to give the following result:

	X	Y	Z
MIN	-1.058	-10.347	503.
MAX	57.262	32.616	589.

and the average value is:

X	Y	Z
28.1	11.1	546.

which compares well with the measured value of:

X	Y	Z
26.6	8.79	538.

These processes give the initial location estimates for the solid structure, which is then used to complete the hypothesis.

Tables 10–5 and 10–6 summarize the results for the primitive ASSEMBLYs in the test images whose estimates arose from using more than one surface. The other primitive ASSEMBLYs had reference frames identical to that of the single surface (rotated into the ASSEMBLY's reference frame if necessary). All results are given in the camera coordinate system. The parameter estimates are good,

Table 10–5: Translation Parameters for Primitive ASSEMBLYS

Surface Name	Test Image	Measured (cm)			Estimated (cm)		
		X	Y	Z	X	Y	Z
robshldbd	1	-13.9	17.0	558.	-15.7	11.5	562.
upperarm	1	0.95	26.4	568.	0.60	17.1	570.
lowerarm	1	26.6	8.79	538.	28.1	11.1	546.

Table 10–6: Rotation Parameters for Primitive ASSEMBLYS

Surface Name	Test Image	Measured (rad)			Estimated (rad)		
		ROT	SLANT	TILT	ROT	SLANT	TILT
robshldbd	1	0.257	2.23	6.12	0.135	2.30	6.28
upperarm	1	3.72	2.23	2.66	3.22	2.24	3.14
lowerarm	1	5.06	2.23	1.32	5.22	2.35	1.18

even considering both the upperarm and lowerarm were substantially obscured.

In the test images, there were several assemblies whose position estimates integrated estimates from subassemblies. They were:

IMAGE	ASSEMBLY	SUBCOMPONENTS
1	upperasm	lowerarm, upperarm
1	robshould	robshldbd, robshldsobj
1	link	robshould, upperasm
1	robot	link, robbody
2	chair	cseat, cbackf, cleg(lf), cleg(lr), cleg(rf), cleg(rr)

The reference frame estimates for these structures are summarized in tables 10–7 and 10–8. All results are in the camera coordinate system. Integrating the different position estimates sometimes gives better results (e.g. cseatf vs chair translation) and sometimes worse (e.g. robbodyside vs robot rotation).

Table 10–7: Translation Parameters for Structured ASSEMBLYS

Surface Name	Test Image	Measured (cm)			Estimated (cm)		
		X	Y	Z	X	Y	Z
upperasm	1	0.95	26.4	568.	0.60	17.1	553.
robshould	1	-13.9	17.0	558.	-15.7	10.3	562.
link	1	-13.9	17.0	558.	-9.7	16.3	554.
robot	1	-13.8	-32.6	564.	-13.5	-35.9	562.
chair	2	-3.5	17.8	427.	4.3	9.2	412.

Table 10–8: Rotation Parameters for Structured ASSEMBLYS

Surface Name	Test Image	Measured (rad)			Estimated (rad)		
		ROT	SLANT	TILT	ROT	SLANT	TILT
upperasm	1	3.72	2.23	2.66	3.20	2.29	3.11
robshould	1	0.257	2.23	6.12	0.135	2.29	6.28
link	1	0.257	2.23	6.12	0.055	2.29	0.05
robot	1	0.0	0.125	4.73	0.0	0.689	4.75
chair	2	6.23	0.469	3.68	6.22	0.789	3.67

Often, there was little effect (e.g. upperasm rotation). In part, the problem is because mapping the subassembly's reference frame expanded it enough (see section 10.2.1) that it only weakly constrained the ASSEMBLY's reference frame. Worse results were obtained when the intersected parameter space had the correct value near one parameter limit, as the average of the limits then drifts away from the true value.

This section has shown how to estimate object 3D orientation using surface shape and substructure placement, as given by the model and 3D surface image data. Better results could probably have been obtained using another geometrical estimate integration method. However, the results here were generally

accurate. This is because of the richness of information in the surface image and geometrical object models.

10.2.2 Feature Visibility Analysis

This section presents results for three topics relating to occlusion, as affecting hypothesis construction. As discussed in section 10.1, the analysis is only applied to individual surfaces, as any structure can be decomposed into surfaces.

Deducing Back-Facing Surfaces

Back-facing surfaces are normally invisible because of their position relative to the viewer. Tangential surfaces are possibly visible, but small errors in estimating the object position make it difficult to determine visibility accurately. If either of these cases are predicted, hypothesis completion does not require image evidence for the surface. Otherwise, it assumes that some image evidence should be visible.

For planar surfaces, deducing this condition is simple: if the predicted surface normal points away from the camera, then the surface is not visible.

Let:

$\vec{n}$ be the model surface normal (by definition is (0, 0, -1))

A be the coordinate transformation from the surface's local system to that of the whole object

G be the transformation from the object's local system to that of the camera.

$\vec{p}$ = a nominal point on the surface in local coordinates

Then:

$\vec{m} = GA\vec{n}$ is the predicted normal
orientation

$\vec{v} = GA\vec{p}$ is the view vector from the
camera to the point on the surface

Test:

if $\vec{v} \circ \vec{m} \geq 0$, then the surface is back-facing

For curved surfaces, this test is augmented to test the normal at each point on the boundary. By the segmentation assumptions (chapter 3), the surface varies smoothly within the boundaries, so if all points on the boundary are back-facing, the interior of the surface must be as well. (When scale considerations are included in future research, this assumption will have to be modified.)

A single front-facing vector is normally enough to declare the surface as front-facing, but a problem occurs with the combination of nearly tangential surfaces and parameter mis-estimation. Here, surfaces predicted as visible may not be so, and vice-versa. This case can be avoided, because it is easy to predict which surfaces are nearly tangential. The test for this is to detect surface normals oriented nearly perpendicular to the line of sight at the surface boundary. The new classification criteria is if a substantial portion of the surface is front-facing, then call it "front-facing". If its not "front-facing" and a substantial portion of the surface is tangential, then call it "tangential". Otherwise, call it "back-facing". That is, the amount of back-facing surface is not important, rather the portion of front-facing or tangential surface determines the classification. The ideal form of this test is:

Let:

T = set of points whose surface normals are
nearly perpendicular to the 3D line
of sight (i.e. the tangential points)

F = set of points whose surface normals face

the viewer, but are not in T (i.e. the front-facing points)

B = set of points whose surface normals face away from the viewer, but are not in T (i.e. the back-facing points)

Then:

If empty(F) and empty(T), then back-facing (i.e. never seen)

If empty(F) and not(empty(T)), then tangential (i.e. possibly seen)

If not(empty(F)), then front-facing (i.e. always seen)

Because of parameter estimation errors, some compromises in the above ideal algorithm were made:

- thresholds were added to decide the visibility class of each vector
- thresholds were added to decide the visibility class of the whole surface

The algorithm to classify individual vectors is:

Let:

$\vec{s_i}$ be the line of sight to point i

$\vec{n_i}$ be the predicted surface normal vector at i

$d_i = \vec{s_i} \circ \vec{n_i}$

Then:

if $d_i > \tau_1$, then $i \in B$ $\quad (\tau_1 = 0.1)$

if $d_i < -\tau_1$, then $i \in F$

$i \in T$ otherwise

The classification of the whole surface is by:

Let:

$$b = size(B)$$

$$f = size(F)$$

$$t = size(T)$$

$$s = b + f + t$$

Then:

if $f/s > \tau_2$, then front-facing ($\tau_2 = 0.1$)

else if $t/s > \tau_3$, then tangential ($\tau_3 = 0.1$)

else back-facing

When this classification was applied to the objects with their estimated reference frames in the two test images, all back-facing and tangential surfaces were correctly deduced. The results are shown in table 10–11.

To summarize, this subsection showed how surfaces could be classified as front-facing, tangential or back-facing by looking at their surface normals relative to the line of sight to the surface.

Deducing Self-Obscured Front-Facing Surfaces

Given the deductions of the previous subsection, all remaining surfaces must be at least partially front-facing. This section describes how partially or wholly self-obscured front-facing surfaces are detected. Here, self-obscured means obscured by other, closer, surfaces from the same object.

The occlusion predictions can be used in three ways:

1. surfaces predicted to be invisible are not searched for,

2. surfaces predicted to be partially self-obscured are verified as having one or more boundaries that show this (e.g. back-side obscuring between this and other object surfaces), and

3. surfaces predicted to be wholly visible are verified as having no back-side obscuring boundaries (unless obscured by unrelated objects).

Table 10-9: Predicted Surface Visibility

Image	Object	Surface Visibility	
1	robbody	front-facing = {robbodyside(1)}	
		tangential = {robbodyside(2)}	*1
1	robshldbd	front-facing = {robshldend,robshould2}	
		tangential = {robshould1}	*1
1	robshldsobj	front-facing = {robshoulds(1)}	
		tangential = {robshoulds(2)}	*1
1	upperarm	front-facing = {uside(2),uends}	
		back-facing = {uside(1),uendb}	
		tangential = {uedges(1),uedges(2), uedgel(1),uedgel(2)}	
1	lowerarm	front-facing = {lsideb,ledgea,lendb}	
		back-facing = {lsidea,ledgeb}	
1	trashcan	front-facing = {tcanoutf(1),tcaninf(1), tcanbot(1)}	
		back-facing = {tcanbot(2)}	
		tangential = {tcanoutf(2),tcaninf(2)}	*1
2	cseat	front-facing = {cseatf(1)}	
		back-facing = {cseatf(2)}	
2	cleg(lf)	front-facing = {clegh}	
2	cleg(lr)	front-facing = {clegh}	
2	cleg(rf)	front-facing = {clegh}	
2	cleg(rr)	front-facing = {clegh}	
2	trashcan	front-facing = {tcanoutf(1),tcaninf(1), tcanbot(1)}	
		back-facing = {tcanbot(2)}	
		tangential = {tcanoutf(2),tcaninf(2)}	*1

*1 – largely back-facing curved surface has tangential sides

Because of parameter estimation errors, tests 2 and 3 are not reliable and are not performed (more discussion below).

The method here uses the object model and position estimates to predict an image of the object, which is then analyzed for visibility. The process occurs in three stages:

1. prediction of visible surfaces

2. deduction of missing surfaces

3. deduction of partially self-obscured surfaces

The first step is implemented using a ray-casting depth image generator. Here, a ray from the viewer is intersected with the model surfaces placed according to the object's estimated position. (The geometrical model makes this easy.) The raycaster produces an array of pixels valued with the depth and identity of the closest (i.e. visible) surface. Figure 10-11 shows the predicted visible surfaces for the trash can superposed over the original image.

The detection of missing structure is now trivial, and consists of finding those front-facing surfaces (from the preceding analysis) not visible in the predicted image.

The detection of partially obscured surfaces is also simple. During the generation of the image, whenever a predicted visible surface pixel was replaced or not included because of a closer pixel, then self-occlusion occurred. The identities of all surfaces that suffered this are recorded during the generation of the synthetic image. Any such surface not completely self-obscured is then partially self-obscured.

Parameter estimation errors may cause nearly obscured surfaces to disappear and barely obscured surfaces to reappear. A similar effect occurs with unobscured surfaces becoming partially obscured (i.e. because a closer surface moves slightly in front) and vice-versa. So, the following algorithm was implemented to decide the visibility classes:

Figure 10–11: Predicted Visible Surfaces for Trash Can

Let:

v = number of visible pixels (predicted by raycasting)

n = number of obscured pixels (predicted by raycasting)

$p = v/(v+n)$ (percentage of visible pixels)

if $p > \tau_1$, then surface is fully visible ($\tau_1 = 0.9$)

if $\tau_1 \geq p > \tau_2$, then surface is partially obscured

($\tau_2 = 0.05$)

Otherwise, the surface is fully obscured

From this analysis, the front-facing surfaces are classified as:

- fully visible,
- partially self-obscured, and
- totally self-obscured.

Table 10–10: Predicted Self-Occlusions

Image	Object	Occlusion Status
1	robbody	fully-visible = {robbodyside(1)}
1	robshldbd	fully-visible = {robshldend,robshould2}
1	robshldsobj	fully-visible = {robshoulds(1)}
1	upperarm	fully-visible = {uside(2),uends}
1	lowerarm	fully-visible = {lsideb,ledgea}
		partially-self-obscured = {lendb}
1	trashcan	fully-visible = {tcanoutf(1)}
		partially-self-obscured = {tcaninf(1)}
		fully-self-obscured = {tcanbot(1)}
2	cseat	fully-visible = {cseatf(1)}
2	cleg(lf)	fully-visible = {clegh}
2	cleg(lr)	fully-visible = {clegh}
2	cleg(rf)	fully-visible = {clegh}
2	cleg(rr)	fully-visible = {clegh}
2	trashcan	fully-visible = {tcanoutf(1)}
		partially-self-obscured = {tcaninf(1)}
		fully-self-obscured = {tcanbot(1)}

The analysis could be extended to show where the occlusion occurs on the partially self-obscured surfaces, and to give a description of the expected visible surface (with boundaries), which could then be compared to the visible surface. This extension was not done.

Table 10–10 records the predicted occlusion status for all front-facing surfaces of all primitive ASSEMBLYs in the two test images. This corresponds exactly with the observed visibility of all surfaces (disregarding external occlusion, which is discussed below). For structured ASSEMBLYs, the process is similar, only some previous cases of external occlusion now become self-occlusion as components are connected together.

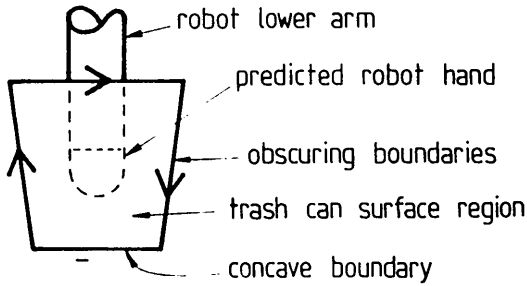


Figure 10–12: Boundaries Surround Completely Obscured Surface

Detecting External Occlusion

Structure obscured by unrelated objects cannot be anticipated in coincidental scene arrangements, unless closer objects can be identified. Hence, though the oriented model predicts a surface is visible, it need not be so. What remains possible is to show that the absence of a feature is consistent with the assumption of occlusion.

If a front-facing, non-self-obscured or partially self-obscured surface cannot be found, then there must be closer, unrelated surfaces completely covering the portion of the image where it is expected. This unrelatedness can be verified by detecting front-surface-obscuring or concave boundaries completely surrounding the closer surfaces, as in figure 10–12.

The next case occurs when front-facing, non-self-obscured surfaces are observed as partially obscured. These must meet all shape and adjacency constraints required by the model and the non-self-obscured invisible portions must

be totally behind other unrelated surfaces (as before). The boundary between the partial object and obscuring surfaces must be obscuring.

The final case of partially externally obscured surfaces which are also self-obscured is not considered here.

Verifying fully obscured structure is the simplest case. Here, every portion of the predicted model surface must be behind another unrelated surface. Minor errors in absolute distance prediction make it difficult to always directly verify that an object surface pixel is further than the corresponding observed pixel, because of parameter estimation errors and nearby surfaces, such as when a piece of paper lies closely on a table surface. Fortunately, relative surface depth differences have already been accounted for in the labeling of obscuring boundaries and the formation of depth-ordered surface clusters (chapter 7). The ordering test can then be reformulated to verify that the entire missing surface lies within the image region belonging to an unrelated, closer, surface cluster . In practice, the test can be performed using a ray-casting technique:

1. Find the set of closer, unrelated surfaces

2. Predict the image locations for the missing surface

3. For each pixel, verify that the observed surface image region has been assigned to one of the closer surfaces

Only the boundary of the missing surface was checked. Obscuring surfaces with holes would invalidate this test, as then the boundary of the obscured surface might be completely hidden, but not its interior.

Again, this ideal algorithm was altered to tolerate parameter mis-estimation:

Let:

P = set of predicted image positions for the boundary of the missing surface

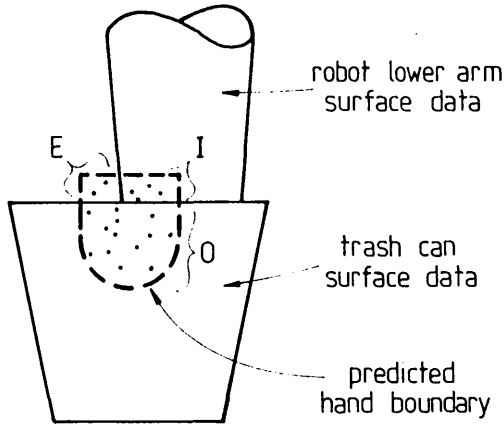


Figure 10–13: Predicted Boundary of Externally Obscured Surface

I = subset of P lying on identified object surfaces
(should be empty)

O = subset of P lying on closer unrelated obscuring
surfaces (should be P)

E = subset of P lying elsewhere
(should be empty)

If:

size(I) / size(P) $< \tau_1$ and

size(E) / size(O) $< \tau_2$

Then: externally obscured ($\tau_1 = 0.2, \tau_2 = 0.2$)

The first condition requires most predicted invisible points not to lie on observed object surfaces, and the second requires most to lie behind closer surfaces. Figure 10–13 illustrates the test.

The criteria for selecting closer, unrelated, obscuring surfaces is:

Let:

S = set of object surfaces

C = set of closer unrelated surfaces

If:

$c_1 \notin S$ and

(a) there is an $s \in S$ such that c_1 and s share an obscuring boundary and c_1 is closer than s

or

(b) there is a $c_2 \in C$ such that c_1 and c_2 share an obscuring boundary and c_1 is closer than c_2

or

(c) there is a $c_2 \in C$ such that c_1 and c_2 share a convex boundary

Then: $c_1 \in C$

Criterion (c) connects obscuring solids together to make complete obscuring surfaces. Concave surface boundaries are ambiguous regarding surface ordering, so are not included. This may cause some failures. Otherwise, figure 10–14 shows a case where an object sitting on a surface with a concave joining boundary would declare the entire background to be potentially obscuring.

The external occlusion analysis discussed above is not fully correct. Figure 10–15 shows surfaces B and C connected by a convex edge, so surface D is behind both. However, surfaces A and B are also convexly connected behind D, but D is in front of A, as well as being both in front of and behind B.

Figure 10–16 shows where a correct case would be rejected because the program could not deduce that B was in front of D. In the absence of more accurate depth predictions, the only correct test may be to observe an obscuring boundary between the visible portions of the object and the missing portions. This less restrictive test was not implemented.

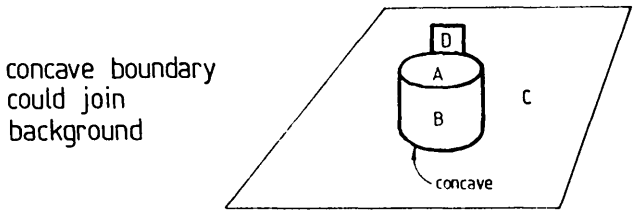


Figure 10–14: Concave Boundary Could Make Background "Obscuring"

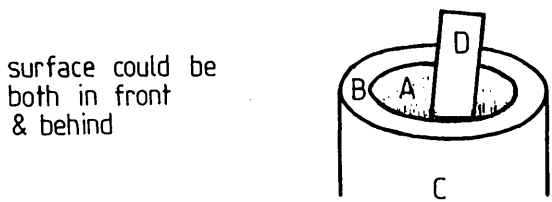


Figure 10–15: Surfaces Could Be Both In Front and Behind

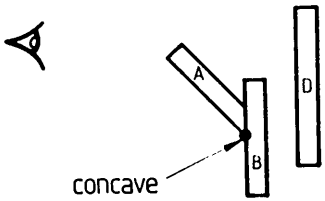


Figure 10–16: No Direct Depth Order Information Available

The only fully externally obscured structure was the robot hand in test image 1, which was correctly detected. Because the reference frame estimates for the lowerarm had a larger rotation angle than was correct, part of the hand was predicted to be not obscured by the trashcan. This motivated for the threshold based test described above.

10.2.3 Direct Evidence Collection

The surface visibility analysis deduced the set of surfaces for which image evidence should be available. This section discusses how such evidence is detected and matched to the model.

Initial (Invocation) Feature Evidence

Surfaces have no substructure, so the evidence for a hypothesized model surface is the associated surface image region. Assemblies are formed by hierarchical synthesis ([TUR74]), so evidence for a hypothesized ASSEMBLY are previously

verified subassemblies or surface hypotheses. This subsection describes how a new ASSEMBLY hypothesis is formed.

If invocation occurs, at least one subcomponent grouping (section 10.2.3) is likely to have positive plausibility, which suggests that particular groupings of subcomponents are visible. Further, some of the image structures in the surface cluster context are likely to have been previously recognized as instances of subcomponents. (If none were invoked, then it is unlikely that the supercomponent will be invoked.) Verified subcomponent hypotheses become the initial evidence for the structure.

For each image structure associated with the invocation subcomponent group, all verified hypothesis of the correct types are located. There may be more than one because of symmetry or ambiguity. Then, groups of these verified subcomponent hypotheses are combinatorially paired with the invoked model's features to create a new hypothesis, provided:

1. Each model feature gets at most one hypothesis, which must have the correct type.

2. No image structure is used more than once.

3. Only maximal pairings are considered.

4. There must be a consistent reference frame that unifies all subcomponents. (This is the initial reference frame calculated as in section 10.2.1.)

The combinatorial matching is potentially explosive, but each image structure generally has only a few identities. Objects with symmetric or duplicated features cause more initial consistent hypotheses, but many of these are eliminated by constraint (2).

Table 10-11 shows the number of initial matches produced for each ASSEMBLY hypothesis in test image 1. While there was considerable opportunity for combinatorial explosion, there were usually only a few potential pairings (those that met criteria (1), (2) and (3)), and even fewer that had a consistent reference

frame (criterion (4)). The greatest number of potential pairings occurred with both symmetric models and symmetric subcomponents.

Additional Surface Feature Location

Given the initial location estimates and the geometrical model, it is easy to predict where a visible surface should appear. This prediction simplifies direct search for image evidence for the feature. This is, in style, like the work of Freuder ([FRE77]), except 3D scenes are considered here. The predictions are taken either from the raycast image or from the oriented model.

In theory, raycasting could predict the exact image area for any missing surface. Unfortunately, parameter estimation variations cause a range of predicted image regions. Figure 10–17 shows the predicted location for the robot upper arm < *uedgel* > panel superposed on the original image using the parameter estimates for < *upperarm* > from tables 10–5 and 10–6. There is a rough agreement on the shape and placement, but the overlap of regions is not a suitable indicator.

To overcome this problem, the oriented model was used to roughly predict where the surface data should appear. Then, because surface data was used, other constraints could be applied to eliminate most inappropriate surfaces from the predicted area. The constraints that a potential surface must meet are:

1. It must not be previously used.

2. It must be in the surface cluster for the ASSEMBLY.

3. It must be in the correct image location.

4. It must have the correct 3D surface orientation.

5. It must have the correct 3D location.

6. It must have the correct size.

Table 10-11: Initial Matches for Each Object in Image 1

MODEL	SURFACE CLUSTER	VALID MODEL	SYMMETRY OF MODEL	DUPLICATE SUBCOMP.	POTENTIAL PAIRINGS	VALID PAIRINGS	
lowerarm	7	Y	2*2	2*4*4	32	2	
lowerarm	9	N	2*2	2	2	1	
upperarm	9	Y	2	2*2	8	2	
upperasm	13	Y	1	2*2	4	4	
robshldbd	3	Y	1	2*2	4	1	*1
robshldsobj	5	Y	2	1	2	1	*1
robshldsobj	2	N	2	0	0	0	*2
robshldsobj	10	N	2	0	0	0	*2
robshould	10	Y	1	1	1	1	
link	15	Y	1	1	1	1	
link	17	N	1	0	0	0	*3
robbody	15	Y	2	2	4	3	*1
robbody	12	N	2	0	0	0	*2
robbody	13	N	2	0	0	0	*2
robbody	8	N	2	0	0	0	*2
robot	15	Y	1	3	3	3	
trashcan	8	Y	2	1	2	2	

*1 - 1 valid pairing lost by parameter space rotation error

*2 - no surface subcomponents found

*3 - duplicate of existing hypothesis

Figure 10–17: Predicted uedgel Panel on Image

7. Its visible portions must have the correct shape.

The implemented algorithm used the constraints 1-5, with some parameter tolerances on 2, 3 and 4. The 7^{th} was not used because likely causes for not finding the surface during invocation were: it was partially obscured, it was incompletely segmented or it was merged during the surface construction process (chapter 6). The result of these would be incorrect shapes. These factors also affect the area constraint (6), so this was used only to select a single surface among any that met the first five constraints (no alternatives occurred in the test images).

The implemented algorithm was:

Let:

S = {all surfaces in the surface cluster not previously used in the hypothesis} = $\{s_i\}$

$\vec{p}$ = predicted image central point for missing surface

$\vec{c_i}$ = observed image central point for s_i

$\vec{v}$ = predicted 3D surface normal at $\vec{p}$

$\vec{n_i}$ = observed 3D surface normal at $\vec{c_i}$

d = predicted depth at $\vec{p}$

z_i = observed depth at $\vec{c_i}$

A = model area for missing surface

M_i = estimated area for s_i

If:

(constraint 3)

$| \vec{c_i} - \vec{p} | < \tau_1$ ($\tau_1 = 20$ pixels)

(constraint 4)

$\vec{v} \circ \vec{n_i} > \tau_2$ ($\tau_2 = 0.8$)

(constraint 5)

$| d - z_i | < \tau_3$ ($\tau_3 = 50$ cm)

Then: s_i is a acceptable surface

The surface selected is the acceptable s_i whose area is closest to that predicted, by minimizing:

$$| 1 - \frac{M_i}{A} |$$

In the two test images, the only missing surfaces were the two side surfaces on the upperarm (image 1) and the inside surface at the back of the trashcan (both images). Visibility analysis for the upperarm showed the side panels were tangential, so no evidence was required for them. For both trashcans, the only potential image surfaces for the trashcan back surface were the two visible surfaces in the surface cluster. The front surface was already used in both cases, and the rear surface met all the other constraints, so was selected.

Rigid Sub-object Aggregation

The preceding analysis was concerned with completing hypotheses using surfaces. However, because of the hierarchical synthesis ([TUR74]) nature of the recognition process, previously recognized sub-objects can be directly integrated as evidence, without having to return to the surface analysis ([FIS83]). Because the sub-object's type is already a strong constraint on its usability, the only remaining constraints are: being in the surface cluster, having the correct adjacent structure and having correct placement. The first criterion selects candidate ASSEMBLYs of the correct type from within the current surface cluster. The placement test is:

Let:

G_s be the global transformation for the sub-object

A be the transformation from the sub-object's to the object's reference frame

Then:

if $G_s A^{-1}$ is consistent with the object's reference frame (see 10.2.1) then allow attachment.

Here, consistent means that the parameter range intersects that predicted by other subcomponents. No structure adjacency criterion was implemented, but subcomponent surfaces should be adjacent to other object surfaces, as conditioned by any external or self-occlusion. Figure 10–7 illustrates the sub-object aggregation process.

Only one instance of a recognizable rigidly connected subcomponent occurred in the two test images. In test image 1, robshldbd and robshldsobj were joined to form robshould. The combination passed the placement test, so proceeded to verification.

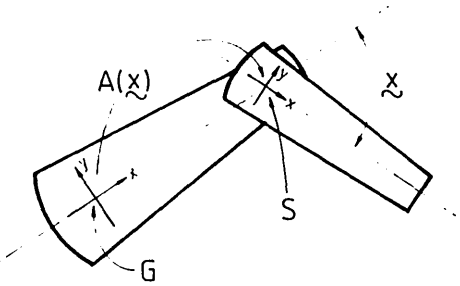


Figure 10–18: Flexibly Connected Subobject Aggregation (in 2D)

Flexible Sub-object Aggregation

Flexibly connected sub-objects, such as the lowerarm ASSEMBLY of a PUMA robot, also need to be aggregated. Here, the flexibly connected subcomponent has been previously recognized, and what remains is to show that its coordinate frame is consistent with the superobject's, given the degrees of freedom inherent in the modeled relationship between their respective coordinate frames. At the same time, the test also binds the values for the remaining degrees of freedom in the coordinate relationship ([FIS83]). This results in numerical values being bound in the particular hypothesis context for the symbolic variables used in the model definition (chapter 5). Figure 10–18 illustrates the flexibly connected sub-object attachment process.

The matching and binding is by a "weak" unification process:

Let:

G be the global reference frame transformation for the object

S be the global reference frame transformation for the subobject

$A(\vec{x})$ be the mapping from the sub-object's reference
frame to the superobject's reference
frame with $\vec{x}$ as the unbound variables

Then:

Compare $G^{-1}S$ to $A(\vec{x})$

Where $A(\vec{x})$ has bound variables, then the values
must match (i.e. parameter estimate ranges overlap)

Where $A(\vec{x})$ has unbound variables, then the
variables are set to the corresponding
parameter ranges from $G^{-1}S$

In test image 1, one binding occurred between inappropriate hypotheses, when constructing the upperasm ASSEMBLY. Each of its two subcomponents (upperarm and lowerarm) had two hypotheses, because of symmetry. Hypothesis pairing produced four pairs, of which three passed the above test (only two should have). Only the results for the correct flexibly connected sub-object aggregations are summarized in table 10-12.

All correct bindings were made, and the table shows that the connection parameters were estimated well. Several of the incorrect combinatorial subcomponent groupings were eliminated during the binding process because of inconsistent reference frames, so this was also a benefit.

10.2.4 Computation Ordering

With such a variety of processes considered under the banner of hypothesis completion, the question of process ordering arises. For example, rather than predict invisible surfaces before detecting surface evidence, the program could explain afterwards why they were not found. Since there may only be a few occluded surfaces, the computational savings may be great. Thus, the reordering of the process sequence may significantly affect its efficiency. This is felt to be an

Table 10-12: Correct Flexibly Connected Subobject

Image	Object	Subobjects	Modeled Parameter	Measured Value	Estimated Value
1	upperasm	lowerarm	jnt3	4.94	4.34
1	link	upperasm	jnt2	2.82	3.07
1	robot	link	jnt1	2.24	2.29
2	chair	cseat	var1	*1	5.89
		cleg(lf)	var2	*1	0.83
		cleg(rf)	var3	*1	0.83
		cleg(lr)	var4	*1	0.72
		cleg(rr)	var5	*1	0.72

*1 – symmetric subcomponent allows any binding

engineering question, however, and no effort was expended on this issue beyond that needed for effective experimentation.

The implemented process sequence for the hypothesis completion was:

1. Find all initial hypotheses using invocation pairings.

2. For each permutation:

   (a) Reference frame assignment:

      i. Estimate coordinate system (may be multiple for surfaces).

      ii. Bind any flexibly connected subcomponents and refine position estimates.

   (b) Image prediction:

      i. Predict back-facing and tangential surfaces.

      ii. Predict fully visible, partially self-obscured and fully self-obscured surfaces.

   (c) Model-directed hypothesis completion

i. Find image surfaces for predicted visible, but uninstantiated, model surfaces subject to constraints.

ii. Find any previously recognized rigidly connected subcomponents.

iii. Find and bind any other previously recognized flexibly connected subcomponents.

iv. Verify missing features as externally obscured.

(d) Fail if any remaining (i.e. unexplained) missing features

10.3 Hypothesis Completion Performance and Discussion

This section presents critical discussion on the results of the chapter. The topics considered are: evaluation criteria, results and criticism of the work with suggested extensions.

Evaluation Criteria and Evaluation

The results that should be generated by the theories in this chapter are clear, and so the evaluation criterion is simply that the implementation performs correctly.

The position and orientation of the objects needed to be estimated correctly. Because of the usual problem with noisy data, the estimates are only approximate; however, the results in tables 10–1 through 10–8 show that the estimation process is accurate.

For each hypothesized ASSEMBLY, analysis deduced each model surface's visibility, given the object's estimated position. Correct surface visibility was predicted for every appropriate model surface, as seen in tables 10–9 and 10–10.

All predicted visible surfaces should have been fully observed and correctly associated, or explained as externally obscured. This was the case for both test

images and the data surfaces provided evidence for the primitive assemblies. All correct rigid and flexible sub-object linkages were made.

Several incorrectly invoked model surfaces were instantiated by similar data surfaces. This resulted in one fully instantiated ASSEMBLY hypothesis – a lowerarm model for the upperarm data (which has a similar shape, as well as was substantially obscured). Given the similarity of the surface shapes, this was reasonable. Several duplicate instantiations resulted from symmetric surfaces or ASSEMBLYs.

Consequently, the hypothesis completion process outlined in this chapter successfully meets its goals.

Criticisms, Improvements and Extensions

The key criticism is over the "idealism" embedded in the matching assumptions. Perfect segmentations at the correct scale was assumed as a prerequisite to matching, whereas this is not particularly realistic. The most stringent criteria – that of all model features being accounted for - is thus probably not ultimately acceptable because position estimate errors will make locating smaller features difficult. Also, segmentation may isolate the desired structures, but at different levels of analytic scale. Hence, some tolerance is needed in the level of evidence at which a hypothesis is accepted. More generally, a full model of an object should have descriptions at several scales and the construction process should match the data across the levels.

The conclusion to this point is that bad or unexpected evidence would have caused failure, as when a surface was too fragmented. The programs accounted for several expected difficulties – as when two surfaces were not properly segmented (as in the two upperarm surfaces in test image 1), or when the thin cylindrical chair legs were too distant to be considered cylinders. Some special case reasoning seems acceptable, but some incompleteness of evidence should also be allowable. Unfortunately, incompleteness leads to requiring match evaluation criteria, or explicit designation of required versus auxiliary evidence.

This work allowed some variation in segmentation by not examining boundary placement when matching surfaces. This avoided some of the problem of segmentation boundary placement as a function of scale. Estimating the spatial rotations for individual surfaces required the boundaries though (during cross-section diameter correlation).

Parameter estimate volumes were successful for coping with data errors. However, the implementation allowed the volumes to expand during map rotations, which resulted in larger error tolerances than necessary. This reduced the effectiveness of using additional data for constraints. Because of this, some failures to recognize symmetric copies of objects occurred when the parameter tolerance range included all possible angles. Hence, better description and manipulation of the estimates is needed.

One possibility is to replace the parameter volume by an algebraic system of constraints (such as in ACRONYM [BRO81]). There is no fundamental difference between the two: a system of algebraic constraints could define every parameter volume, and a (perhaps unbounded) parameter volume could define every algebraic constraint. Systems of constraints are equivalent to the (possibly void) intersection of the parameter volumes. The algebraic formulation would allow easier constraint definition and incremental introduction of constraints as evidence accumulates. The problem with such a method is the difficulty in discovering solutions to the (possibly nonlinear) set of constraints. The parameter intersection method is easier to solve for values, but becomes computationally expensive when other than position parameters are included because of the high dimension parameter spaces.

Another major criticism is that the recognition process only uses surfaces. The traditional "edge" is still useful, especially as surface data does not represent reflectance variations (e.g. surface markings). Volumetric evidence could also be included. Absolute surface reflectance is also relevant. Relationships between structures, such as line parallelisms and perpendicularities can provide strong evidence on orientation, particularly when occlusion leaves little visible evidence.

The validation of the externally obscured components suffers from being un-

able to always correctly determine when unrelated surfaces are potentially obscuring, especially when coupled with problems of parameter estimation. More work is needed here. One possibility is to use more topological knowledge, such as surface ordering, because of its greater reliability. If a feature is obscured, then there will be an obscuring boundary somewhere between the feature's predicted position and the rest of the object.

A minor problem was that visibility analysis of the robot upperarm in test image 1 declared all four side panels as tangential. Though this was correct, two panels were clearly visible and this evidence should have been used. This would also have required some new reasoning, as the two panels were merged into a single surface because the post obscured their common boundary.

A general problem was that there were many thresholds used in the different processes, and some of these are likely to be scale dependent. Most of the thresholds arose because of prediction uncertainities affected by position estimate uncertainities. Further analysis might permit more general discrete reasoning to avoid this approach.

Super-object knowledge could help recognize sub-objects. For flexibly connected sub-objects, each sub-object is currently recognized independently and then aggregated in a strictly bottom-up process. However, one sub-object may invoke the object, which could partially constrain the identity and location of the other sub-objects. Since these objects often obscure each other in unpredictable ways, there may not be enough evidence to independently invoke and identify a sub-object, whereas additional active super-object knowledge might overcome this.

While hypothesis construction does look for missing *surface* evidence, missing *substructures* lead to failure. This is a weakness of the implementation. One improvement would be to do better occlusion analysis based on currently held hypotheses (i.e. predict what aspects of the sub-object might be visible if the super-object is at a given position). The expedient solution used here is to invoke subcomponents before components, the order being determined by the model hierarchy.

Research Contributions

The most important contribution of this chapter is the investigation of mechanisms for using surfaces as the primary recognition evidence. Faugeras and Hebert ([FAU83]) have also demonstrated how to use surfaces, but their approach, while using real data, suffered from not being able to account for all present data, nor for the disposition of all model features. This chapter showed how to use models, surfaces and associated positional information to:

- estimate the reference frame for objects,
- deduce the visibility of all model features,
- predict where to find all visible features,
- explain missing data as instances of occlusion, and
- ensure consistent data.

In particular, reference frame estimates with a surface-oriented object model allowed prediction of which surfaces were fully visible, tangential, back-facing, partially obscured or fully obscured. Those visible surfaces not detected were assumed to be obscured by external objects, and some conditions for verifying this case were demonstrated. The surface cluster formulation was shown to be useful for providing the contexts within which to locate image evidence.

The chapter also demonstrated several new techniques for direct position parameter estimation. Single surfaces had their orientations estimated using the nominal surface normal and their boundaries. An alternative for curved surfaces used the surface normal and curvature axis. Solids had their orientation estimated by solving for the rotation that mapped pairs of model vectors to data vectors.

Finally, the chapter showed how the hierarchical synthesis process could be adapted to surface-based object models. Also, it was shown how to combine

flexibly connected subcomponents and bind the variables of flexibility at the same time.

Chapter 11

Hypothesis Verification

The model invocation and hypothesis construction processes are based on matching descriptions of the substructures. Consequently, it is possible for coincidental scene arrangements to lead to spurious object hypotheses. Many of these hypotheses will have been eliminated at earlier stages in the processing, but some may remain. This chapter discusses some additional constraints on solid physical objects that help guarantee object existence and identity.

11.1 What Should Verification Do?

All object recognition is based on matching perceptual to model features. Unfortunately, the perceptual features are not always correct interpretations of the underlying physical phenomenon (e.g. a surface orientation edge may have been called a reflectance edge by an ignorant process). Further, the matching process may have been successful even though the resultant object was fictitious, owing to coincidental arrangements of the scene objects.

Models are invoked by attributes suggesting objects, and invocation is thus necessarily coincidental. Construction is more constraining, requiring geometrical coordination among features as dictated by the model, but this can still leave spurious, well advanced, hypotheses that need to be somehow eliminated. Hence:

verification attempts to ensure that what is recognized is only what is contained in the scene.

So, verification has two purposes. Practically, it eliminates hypotheses that arise from coincidental arrangements of image features. Its more philosophical purpose is to maximally confirm the validity of the hypothesized identity of an image structure, to the limits of the object representation. The point is to ensure both the physical existence of the object and its having the requisite properties, by extending the depth of the subordinate concept structure beyond merely superficial attributes. In a sense, this is a true "seeing" of the object. Previous stages of analysis consider only subsets of the features in a "suggestive" sense, whereas verification looks for all features and can report what it finds.

In practice, there are a set of constraints that an object must hold in order to be said to exist and have a given identity. Verification ensures that these constraints are satisfied. This entails knowing both what is important in an object and its representation, and what makes it appear as it does.

The input to verification is a fully instantiated object hypothesis. As discussed in chapter 10, the hypothesis construction process understands occlusion and records missing structures whose location was predicted, but which were declared obscured based on other evidence. Verification of partially obscured structures must show that the remaining visible portions of the object are consistent with what is predictable given the model, its spatial location, the occlusion annotations and the image evidence.

We would like reasonable criteria for ensuring correct object hypotheses, with "reasonable" encompassing both richness of detail and conceptual appropriateness. Unfortunately, as briefly discussed in chapter 4, "ensuring" is impossible because all interpretations of sensory data are necessarily imperfect and because no object can be completely and uniquely characterized. Practical problems are related and stem from lack of resolution in the data and impoverished models and descriptive terms. However, some verification is obviously both necessary and

of value and is intended to remove the most obvious cases of mis-identification. Many practical advances remain, some of which are attempted here.

Verification could be achieved by reproducing the image from the object hypothesis and then doing detailed surface comparisons, but this is both computationally expensive and unnecessary. Verification should be carried out at a conceptual level that is efficacious, representationally appropriate, and efficient.

Existence Verification

Verification can assume that surfaces exist as they are data primitives. Assembly existence requires the objects to be completely bounded by a connected set of surfaces.

Object identification has been predicated on detecting discriminating features, but little work has addressed the question of: "Does the object exist, or are the features merely coincidental?". Geometrical constraints (ACRONYM ([BRO81])) increase the certainty by showing that the feature's image positions are consistent with the 3D location of a particular object instance. What is desired is to show that these features are causally related. But, because of philosophical and practical difficulties, the work will consider instead how to eliminate assembly's whose features are definitely unrelated (in the context of that assembly).

Figures 11–1 through 11–3 show three orthogonal planes in three different configurations, each of which invokes a cube model, but fails to meet the general physical requirements of: (a) connectedness, (b) closure (e.g. all structures meet properly) and (c) minimality (e.g. no extra structure is found). Cases (b) and (c) are related in that one is equivalent to the other when seen from another viewpoint. The distinction is that different evidence is used to detect the two cases.

Existence verification becomes more difficult in the context of generic objects, because the more abstract the definition becomes, the fewer data features are matched to model features. Hence, the cases shown in the figures become

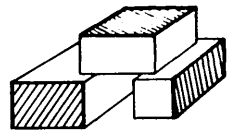


Figure 11–1: Unrelated Planes Invoke Cube

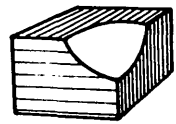


Figure 11–2: Related Planes with Internal Gap Invoke Cube

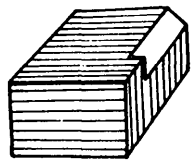


Figure 11–3: Related Planes with Unaccounted-for Structure Invoke Cube

allowable physical objects at the generic level. Another example occurs if the legs of a real chair are not modeled as part of a generic chair (which might have only a seat patch isolated from a back patch). One resolution of this dilemma may be to require explicitly generic models to indicate their status and so allow bypassing of these tests.

Identity Verification

Verifying the existence of an object does not guarantee that it has been identified correctly. Identity verification is needed because several similar models could be invoked and successfully constructed. This suggestive "recognition" is fine for artistic vision systems, but inappropriate for precise object recognition, which is the intent of this research.

General physical properties are important to existence and efficient representation, but the uniqueness of identity is dependent on individual object properties. A spheroid, an ellipsoid and an "egg-shape" are structurally similar; all are

valid physical objects and are likely to have the same surface segmentation, but are obviously discriminable using concepts like symmetry. More refined data is needed for identifying one's car among similar ones by a small dent in the door. Hence, object-instance-specific as well as object class-specific properties are needed to discriminate between alternative identities.

It would be ideal if all object features could be considered, instead of merely a discriminating subset. Unfortunately, it is unclear at what level to represent these features. A constraint on the area of a rectangular surface could also be expressed as constraints on the two dimensions. Because this research has concentrated on object shape the discriminating properties used are based on shape, though it is unclear whether this should be done on a point by point basis or by characterizing segmented regions.

General physical constraints clearly play an important role in limiting both the types of hypotheses satisfying a set of data and the types of information needed to characterize real objects. The objects considered in this thesis are flexibly attached rigid solids defined by characterizable surfaces (with the possibility of two-sided laminar surfaces), and the constraints used reflect this. In the next section, specific constraints will be proposed for each of the classes of objects (surfaces, rigidly connected solids, flexibly connected assemblies), but for now general problems will be considered.

Shape is the basis for surface identity. The segmentation assumptions imply that surface class, curvature parameters and boundary location are sufficient to define the surface, and are all that need to be compared. Comparison requires knowing the surface's 3D position and which observed boundaries correspond to model boundaries (as distinct from obscuring or tangential boundaries). Because position estimates may be slightly erroneous, detailed boundary and surface comparison is inappropriate: more global comparisons are needed.

Verifying assembly identity is more difficult, because subtle differences in shape are not yet easily discriminable. Disregarding this problem, *identity is maximally verified if all the predicted visible features are found in the correct places.* The subcomponent identities were verified earlier in the hierarchical

synthesis process. Verifying their configuration requires checking that the subcomponents have the correct spatial location and orientation in the reference frame of the whole assembly, and that their connections correspond to those given by the model.

Flexibly connected structures have had their subcomponents previously verified, so what remains is to validate their relative configuration. Otherwise, all the correct, but unassembled, components could be identified as the whole assembly. This test requires knowing the relative coordinate frame relationships, to see if appropriate structures are aligned and flexible attachment parameters are in the correct range.

Again, as certainity of identity is impossible, the goal of identity verification is instead to falsify hypotheses not meeting all identity constraints.

In summary, verification must:

1. cover the three classes of structures,
2. question both existence and identity,
3. verify both shape and configuration, and
4. use boundary, surface shape class, surface adjacency and relative reference frame information.

11.2 Constraining Object Existence and Identity

The previous section concluded that verification must try to ensure that the hypothesized object exists and that its identity is correct. The surface, the rigid assembly and the flexible assembly have separate verification requirements which will be individually considered in the subsections that follow.

Existence is based on general object properties. Surfaces are presumed to exist as inputs to recognition, so only ASSEMBLY existence needs be verified. The goal is to reject hypotheses that are coincidental, which means showing that the surfaces associated with the hypothesis cannot be organized into a solid. Solidity is based on complete connection of all visible surfaces, which requires a topological examination of the evidence.

Identity is based on object-specific properties. Associated with each model is a set of constraints that the data must satisfy, and any structure that meets these is called an instance of the model. Any hypotheses that do not meet the constraints are rejected.

Identification is complete to the level of description embodied in the model. In the reviewed research, the level of detail for most 3D object recognition was superficial and so an object meeting the criteria was identified as far as the computation was concerned, but, unfortunately, not for us as observers. The solution is to increase the level and structure of the evidence.

There is a question of how constraints on properties should be expressed. ACRONYM ([BRO81]) implemented some property requirements using numerical constraints on the object's attribute values (e.g. the image angle between the spines of two "ribbons" is in a given range). Each property gives some measure of certainty and cumulatively constrains the possible objects to eventually ensure identification. A flying bird would probably be recognized as an example of an airplane, given ACRONYM's models, but more properties would discriminate between the two. (It is reasonable for a bird to invoke the airplane model, though.)

While the property value approach is useful for initial identification of the objects, the goal here is to reject false hypotheses, and this is assumed to require comparison between the object and model surface shapes as these are the primary visible features of the two. A first approach to comparing surfaces can be made by comparing the symbolic characterizations of the surfaces (the boundaries, the curvature axes and the curvature magnitudes) and the relative relationships of these features in the object reference frame.

11.2.1 Surface Verification

Invocation of surfaces is based on summary characteristics (e.g. areas), rather than shapes. Hence, verification compares shapes. As surface regions are characterized by their boundaries and internal shapes, the verification ensures that:

- The image surface has the same shape as that of the forward-facing portions of the oriented model surface.

- The surface image boundaries are the same as those on the forward-facing surfaces predicted by the oriented model.

This entails determining which boundaries are visible and adding new ones when the surface is tangential to the line of sight.

Implicit in these tests is the knowledge of the location and orientation of the model surface.

There are two sources of problems that complicate the above criteria: inexact boundary placement at surface curvature discontinuity boundaries, and information loss because of occlusion. (There are other problems related to placement and identification of boundaries as a function of scale, but these are ignored here.)

The first problem causes variable sized surface regions and hence makes it difficult to directly compare surfaces and boundaries exactly. But, some possibilities remain. In particular, all model boundaries are either orientation or curvature discontinuity boundaries. The former should remain stable because the effects are more dramatic than at the latter and should appear as either shape segmentation or front-side-obscuring boundaries, whose locations are predictable. Detailed shape analysis can distinguish front-side-obscuring boundaries arising from orientation discontinuities and those arising from being a generator on a tangential surface. Curvature discontinuity boundaries are weaker and should perhaps be ignored.

Occlusion causes data loss, but this event is detectable as the back-side-obscuring boundaries associated with the surface indicate the initial point of occlusion. Concave boundaries are also ambiguous regarding surface ordering, so need not be true surface boundaries. Hence, boundary comparison only applies to the visible portions of the original surface region, and not to back-side-obscuring or concave boundaries. Similarly, surface comparison can apply only to the visible portions of the surface. A further point is, as the visible data must be a subset of the predicted data, the back-side-obscuring boundary must be internal to the predicted surface. Figure 11-4 illustrates these points, whose criteria are:

[S_1] All data boundaries labeled as front-side-obscuring and surface orientation discontinuity should closely correspond to portions of the boundaries predicted by the model and those labeled as curvature discontinuity approximately so. The back-side-obscuring boundaries must be internal to the predicted region.

[S_2] The data surface should have the same shape as a subset of the surface of the oriented model, except where near curvature discontinuities.

Because of errors in estimating surface reference frames, it was difficult to predict surface orientation and boundary locations accurately enough for direct comparison. Hence, the only comparisons made here are surface curvature and axis orientation. These are more reliable because they are integrated shape properties. If the estimates had been better, approximate shape comparisons could have been performed after translating the predicted surface to get a good overlap with the data surface.

As a result, only test S_2 was implemented:

[S_2] Surface Shape Verification Test

Let:

S and $\hat{S}$ be the predicted and observed surface shape class

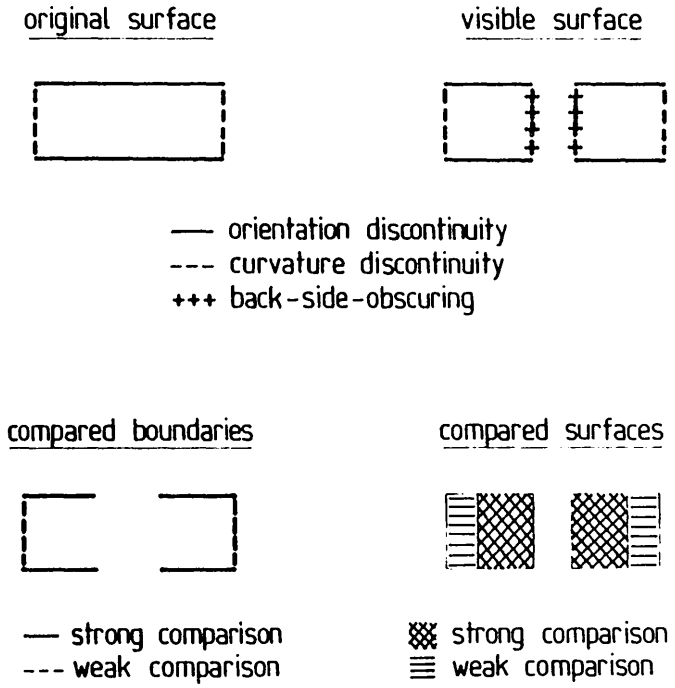


Figure 11–4: Boundary and Surface Comparison

M and $\hat{M}$ be the predicted and observed
major surface curvatures
m and $\hat{m}$ be the predicted and observed
minor surface curvatures
$\vec{p}$ and $\vec{a}$ be the predicted and observed
major curvature axis vectors
τ_c, τ_a be thresholds

If:

S is the same as $\hat{S}$,

$| M - \hat{M} | < \tau_c$, $(\tau_c = 0.05)$

$| m - \hat{m} | < \tau_c$, and

$| \vec{p} \circ \vec{a} | > \tau_a$ $(\tau_a = 0.80)$
(planar surfaces do not use this last test)

Then: the surface passes the test

11.2.2 Rigid Assembly Verification

Rigid assemblies have to meet both the existence and the identity requirements discussed previously. The existence requirements are necessary because the surfaces constituting the constructed hypothesis correspond to the surfaces of the hypothesized object, but are not guaranteed to make up a true object.

Most real objects are compact solids and one manifestation of this is continuity of surfaces connecting all components of the object. The criterion for this property is that all surfaces composing the object must somehow be connected to each other with no extra material. Intuitively, this property creates a web of connecting surfaces that delineate the object, and so eliminate hypotheses formed

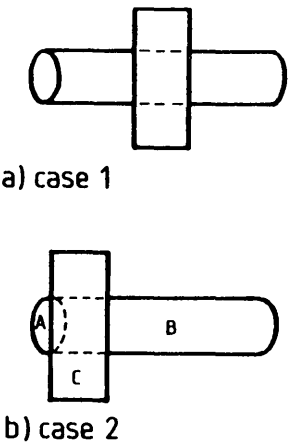


Figure 11–5: Surface Adjacency Behind Obscuring Structure

from coincidental arrangements. Figure 11–1 showed three unrelated planes that lead to a cube hypothesis that fails this criterion. Obscuring objects can disrupt this connectedness in the image, but the surface reconstruction process (chapter 6) eliminates some cases of this. Figure 11–5 part a illustrates this. Some cases, like that in figure 11–5 part b are not solved by this and so a second rule is also used: If both surfaces A and B are behind the same obscuring surface cluster (C), then A and B could be adjacent.

Because of self-occlusion, direct connections may not be visible. Further, concave boundaries are ambiguous regarding surface connectivity. Therefore, it

is difficult to determine if two surfaces are directly or indirectly connected, and hence if the whole proposed assembly is connected. So, instead, hypotheses will be rejected if it is certain that they cannot be fully connected. Then, a hypothesis can be rejected if there are subcomponents between which no conceivable connection exists. For this analysis, it is assumed that background surfaces are those that touch the image boundary or are behind all other surfaces.

The implemented test is:

[E_1] All Surface Hypotheses are Potentially Connecting

Let:

$\{S_i\}$ be all non-background data surfaces
$\{D_j\}$ be all data surfaces used in the hypothesis
$PC(S_a, S_b)$ hold if S_a and S_b share any type of boundary
(i.e. are adjacent and therefore potentially connecting)
$TC(S_a, S_b)$ be the transitive closure of $PC(S_a, S_b)$

If:

for some D_a and D_b, $TC(D_a, D_b)$ does not hold

Then: the hypothesis is incorrectly formed

Intuitively, correct object identification is assumed if all the right structures are found in the right places. Given the connectivity guaranteed by the existence of the object, merely having the correct components is likely to be sufficient because the surface shapes of most objects only fit together rigidly and completely in one way (disregarding highly regular objects, like blocks). The requirement of consistent reference frames will eliminate many arbitrary groupings. But, because there are likely to be a few counter-examples, especially with symmetric objects and potential mis-identifications of similar surfaces, geometrical as well as topological correspondences are required.

For rigid objects, the essence of identity is shape, and surface images make this information directly available. Given the surface image, the observed shape

could be compared to that of each object from each viewpoint, but this approach is computationally infeasible. A more parsimonious solution follows, which also considers weak segmentation boundaries and occlusion.

Surfaces that are connected according to the model should be connected in the scene. This does not always imply adjacency is maintained, because objects are three dimensional, and boundaries are not visible from all viewpoints. As discussed in section 11.1, extra unaccounted-for structure attached to the object must be allowed for generic models. Because generic models need not have all modeled surfaces connected, the data surfaces matched to them need not connect either. The conclusion of these points is that some surface adjacency constraints can be applied, but they must be weak.

There are cases where both adjacent surfaces are visible but the common boundary is invisible behind the object. Here, the surfaces must pass from being front-facing, to adjoin on the back side. Hence, they will be classified as being tangential and so any tangential surfaces will not be tested.

It is assumed that segmentation produces the correct regions, but the position of surface curvature discontinuity boundaries may vary slightly, which is the same segmentation condition considered for surface verification.

Occlusion reduces the available information. In making the model-to-image pairings, the hypothesis completion process reasoned about partially and fully obscured surfaces and noted when these instances occurred. Occlusion affects verification because some surfaces may be partially or completely missing or a surface may be broken up by closer surfaces, hence correspondences and comparisons will be affected. True surface boundaries may be obscured. The remaining true surface boundaries will be connected by back-side-obscuring boundaries in different locations. Since these are not model features, they are ignored.

Parameter estimation errors may also reveal or hide nearly tangential surfaces, so variation must be allowed.

Based on these ideas, the rigid object identity constraints are:

$[R_1]$ – Each visible forward-facing model surface can have at most one data surface paired with it. Each data surface can have at most one visible forward-facing model surface paired with it.

$[R_2]$ – the shapes of the unobscured forward-facing portions of model surfaces are the same as those of the corresponding data surfaces, except when near a curvature discontinuity segmentation boundary.

$[R_3]$ – the position of observed image surfaces relative to each other is as predicted for the corresponding model surface.

$[R_4]$ – model surface adjacency implies data surface adjacency. The reverse test of ensuring adjacent data surfaces have corresponding model adjacency is not applied because generic models may not connect model surfaces.

These constraints are implemented as the following tests:

Let:

$\{F_i\}$ be the visible forward-facing model surfaces

$\{I_i\}$ be the image surfaces

$\vec{P}_i$ and $\vec{O}_j$ be the predicted and observed center-of-mass for the corresponding model and image surfaces F_i and I_j

$\vec{N}_i$ and $\vec{M}_j$ be the predicted and observed surface orientations at the centers-of-mass for the corresponding model and image surfaces F_i and I_j

τ_t and τ_r be thresholds

Then:

$[R_1]$ For each I_i there is at most one corresponding F_j.
For each F_j, there is at most one corresponding I_i.

$[R_2]$ Individual surfaces are already verified as in section 11.2.1.

Surfaces added during hypothesis completion are not verified under this rule because they are likely to be inaccurate (i.e. partially obscured).

$[R_3]$ For each corresponding I_i and F_j:

$$| \vec{P}_i - \vec{O}_j | < \tau_t \qquad (\tau_t = 20.0)$$

$$| \vec{N}_i \circ \vec{M}_j | > \tau_r \qquad (\tau_r = 0.8)$$

$[R_4]$ Let:

I_a, I_b be two non-tangential data surfaces

F_a, F_b be the corresponding model surfaces

If:

F_a and F_b are adjacent in the model,

I_a and I_b are not adjacent in the data, and

(a) I_a is not partially obscured,

(b) I_b is not partially obscured

or

(c) there is no group of surfaces partially in front of both I_a and I_b,

Then: the hypothesis is incorrectly formed

These tests try to ensure identity by showing all the right structures are in the right places. Surface verification showed individual surface identities were plausible, and these tests show the assembled whole is also. Test R_1 verifies that the predicted structures are found, R_2 that the local orientation and shape of the surfaces are as predicted, R_3 that the relative position is as predicted and R_4 that surface connectivity is correct.

Occlusion also has distinctive characteristics, and thus the hypothesis that an object is partially or fully obscured should be subject to some verification. Back-side-obscuring boundaries usually signal this occurance, though not al-

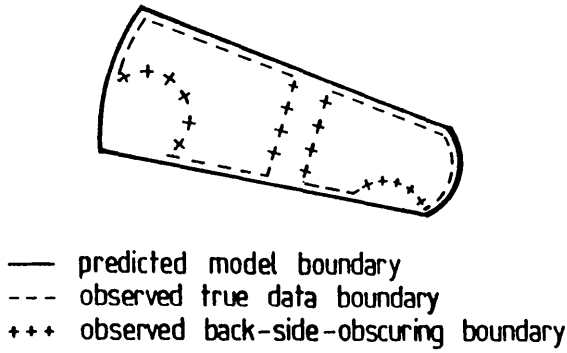


Figure 11–6: Occlusion Boundaries Lie Inside Predicted Model Boundaries

ways. When a curved surface goes from facing the viewer to facing away, self-occlusion occurs without back-side-obscuring boundaries. When back-side-obscuring boundaries are present, though, three new constraints can be added:

[O_1] – the back-side-obscuring boundary should lie inside the image region predicted for the surface. Alternatively, the predicted image boundary should lie on or outside the observed image region. Figure 11–6 illustrates this.

[O_2] – Back-side-obscuring boundary segments that bound the surface image region must end as the crossbar of a "TEE" junction. This implies that there must be at least three image regions at the junctions. Figure 11–7 illustrates this.

[O_3] – A non-tangential image surface should be predicted as partially self-obscured during visibility analysis (chapter 10) iff the corresponding data surface has at least one back-side-obscuring boundary whose closer surface is also an object surface.

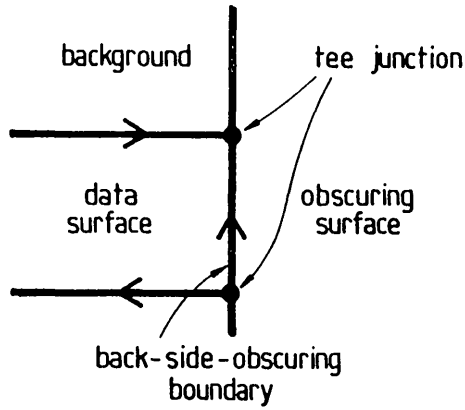


Figure 11–7: Occlusion Boundaries End on TEEs at Surface

Constraint O_1 is not applied because parameter estimation errors currently make it difficult to check this condition reliably (e.g. predicted model and data surfaces do not overlap adequately). Constraint O_2 is guaranteed assuming image labeling is correct, which is the case here. When automatic segmentation is used, then this constraint could help ensure consistency.

Because of parameter estimation errors, it is likely that there are self-occlusions predicted during raycasting that are not observed (because of surfaces becoming slightly obscured). Hence, the test of verifying predicted self-occlusions was not performed. While it is also possible for the reverse to occur (i.e. for slightly obscured data surfaces to be predicted as not obscured), it was felt that if a self-occlusion was significant enough to be observed in the data, then prediction was likely to show it even with parameter estimation errors. Hence, the reverse test was implemented: if a data surface is observed as partially obscured by another data surface used in the hypothesis, then the model must predict this.

The test for O_3 is:

[O_3] Observed Self-obscured Surfaces are Predicted

Let:

$\{D_i\}$ be the non-tangential partially obscured data surfaces

$\{C_{ij}\}$ be the closer data surfaces across obscuring boundaries around D_i

S_i be the model surface corresponding to D_i

$\{M_k\}$ be the other model surfaces

front(X, Y) hold if model surface X is directly or indirectly in front of X. This is found by raycasting and taking the transitive closure.

If:

For each D_i and each C_{ij}
If there is a M_k corresponding to C_{ij},
then front(M_k, S_i)

Then: the self-occlusion is as predicted by the model.

One application of O_3 was particularly significant. The robot upper and lower arms are nearly symmetric, so there are two values for the upperarm position and joint angle where the lowerarm can be nearly in the position shown in test scene 1 (disregarding self-occlusion for the moment). The difference between the two cases is whether the lowerarm is in front of the upperarm or behind. Though the depths of the component reference frames are different in the two cases, parameter tolerances do not completely reject the second alternative. Fortunately, test O_3 does discriminate.

11.2.3 Flexible Object Verification

Flexible object verification is trivial in comparison to the previous structures. By virtue of the hypothesis construction process, all subcomponents have been previously verified. Further, because of the coordinate frame matching process, the reference frames of the subobjects must have the appropriate alignment relationships with the whole assembly. What remains is to verify that the variable parameters meet any given constraints. These are constrained using the general method given in the next subsection

11.2.4 Numerical Constraint Evaluation

Many numerical values are associated with hypotheses. The most important of these are the property values described in chapter 8, but could also be other values such as the object position or joint angles. Constraints can be specified on these values. All constraints relating to the structure being verified must hold for the validation to succeed.

The constraints were mainly used for eliminating spurious surface hypotheses and usually tested absolute surface area.

The constraints are specified as part of the model definition process, as a set of statements of the form:

CONSTRAINT< *name* > < *constraint* >

The < *constraint* > must apply in the context of structure < *name* >. Here:

< *constraint* > ::= < *pconstraint* >
|< *constraint* > AND < *constraint* >
|< *constraint* > OR < *constraint* >
|(< *constraint* >)

< *pconstraint* > ::= < *value* >< *relation* >< *number* >

< *relation* > ::= <|>|=|<=|>=| !=

< *value* > ::= < *variable* >|< *property* >(< *name* >)

The < *value* > refers to a variable or a property (possibly of a substructure) in the context of the structure being constrained. Other constraint expressions could have been easily added. The verification of these constraints is trivial.

An example of such a constraint for the elbow joint angle jnt3 in the robot upperarm assembly is:

CONSTRAINT upperasm (jnt3 < 2.5) OR (jnt3 > 3.78);

which constrains the joint angle to 0.0 – 2.5 or 3.78 – 6.28. Another constraint is:

CONSTRAINT uside ABSSIZE(uside) < 1900.0

which constrains the absolute surface area of uside to be less than 1900 cm^2.

11.3 Verification Performance And Discussion

This section evaluates the theory given in the previous section, as based on the example images in appendix A. It presents the evaluation criteria, discusses performance on the test images, criticizes the results, makes some suggestions for improvements and the summarizes the contributions of this chapter.

Evaluation Criterion

The ideal criterion is whether verification accepts all and only the true instances of objects. However, as tolerances are needed to allow for segmentation variations, position parameter mis-estimation, and obscured surface reconstruction,

some invalid verifications are expected. Some invalid surfaces are verified because of variability in surface shape matching and having no other constraints on their identity at this point. The effect of these hypotheses is reduced performance rates and increased chances of invocation of higher level false objects. However, verified higher false hypotheses are not likely to occur as the surfaces must then meet grouping, relative orientation and location constraints in hypothesis construction, and the verification constraints discussed in this chapter.

The most important criterion for verification is the contrary: no true objects should be rejected. So, the proposed evaluation criteria are:

1. no true hypotheses are rejected, and

2. among false hypotheses, only low level (e.g. surfaces), symmetric or ambiguous hypotheses are accepted.

Evaluations

Some false ASSEMBLY hypotheses were rejected in hypothesis completion because no consistent reference frame could be found for them. These hypotheses are included in the analysis of rejected hypotheses given below (along with rejections by the criteria given in section 11.2).

Table 11–1 summarizes the causes for rejection of surface hypotheses in the test images. Some rejected curved surface hypotheses had the correct identity but an inconsistent reference frame. Table 11–2 summarizes the causes for rejection of assembly hypotheses. Table 11–3 lists and analyzes all remaining verified hypotheses that were not "correct". The table records the rejection criterion as given in section 11.2, except for those designated by "N", which means rejection by a modeled numerical constraint (section 11.2.4), by "H", which means failure to establish a reference frame (chapter 10), or by "A" which means all slots that should have been filled were not.

By the results shown in the three tables, verification works well. Two true assembly hypotheses were rejected because of deficiencies in other modules rather

Table 11–1: Surface Hypothesis Rejection Summary

TEST IMAGE	MODEL	IMAGE REGIONS	REJECTION RULE	INSTANCES
1	uside	12	N	1
1	uends	25	S_2	2
1	uendb	19,22	N	2
1	lsidea	19,22	N	1
1	lsideb	19,22	N	1
1	lendb	25	S_2	2
1	robbodyside	9	N	4
1	robbodyside	8	S_2	2
1	robshould1	12	S_2	1
1	robshould2	12	S_2	1
1	robshoulds	27	S_2	2
1	cbackf	9	S_2	2
1	tcanoutf	9	S_2	1
1	tcaninf	9	S_2	2

Table 11–2: Assembly Hypothesis Rejection Summary

TEST IMAGE	MODEL	IMAGE REGIONS	REJECTION RULE	INSTANCES	NOTE
1	lowerarm	12,18,31	H	30	
1	lowerarm	17,19,22, 25,32	A	1	
1	lowerarm	17,19,22, 25,32	H	1	
1	upperarm	17,19,22, 25,32	H	6	
1	upperasm	12,17,18, 19,22,25 31,32	R_3	2	
1	upperasm	12,17,18, 19,22,25 31,32	O_3	1	
1	robshldbd	16,26	H	3	
1	robshldsobj	29	H	1	*1
1	robbody	8	H	1	*1
1	robot	all appt.	H	2	

*1 – failure because of parameter rotation error

Table 11-3: Other Verified Hypotheses Analyzed

TEST IMAGE	USED MODEL	TRUE MODEL	IMAGE REGIONS	NOTE
1	uside	uside	19,22	*3
1	uends	uends	25	*2
1	lsidea	lsideb	12	*1
1	lsideb	lsideb	12	*2
1	ledgea	ledgea	18	*2
1	ledgeb	ledgea	18	*1
1	lendb	lendb	25	*2
1	lendb	uends	31	*1
1	robbodyside	robbodyside	8	*2
1	robshould1	robshould2	16	*1
1	robshould2	robshould2	16	*2
1	lowerarm	lowerarm	12,18,31	*2
1	upperarm	upperarm	17,19,22, 25,32	*2
1	robbody	robbody	8	*2
1	trashcan	trashcan	9,28,38	*2

*1 - true model similar to invoked model

*2 - symmetric model gives match with another reference frame

*3 - error because substantially obscured

than failing verification requirements. All verified false hypotheses were reasonable, usually arising from either a similar or symmetric object model. Most rejected surface hypotheses failed the value constraint (usually surface area – see appendix B). Curved surfaces were rejected when their curvature axis was inconsistent with other orientation estimates. Most assemblies were rejected because no consistent reference frame could be found. (Many of these hypotheses arose because hypothesis completion has a combinatorial aspect during initial hypothesis construction.) The other major rejection criteria were:

- incomplete hypothesis (A)
- surface shape or orientation inconsistencies (R_3)
- self-occlusion inconsistencies (O_3)

Assembly existence criteria failed no hypotheses, perhaps because the surface cluster context for hypothesis completion strengthened surface relations.

Criticisms and Areas for Improvement

The most important deficiency of verification is its dependence on a literal model of the objects recognized. Objects are probably more suitably recognized and confirmed by combinations of desirable properties and the absence of any undesirable ones, than by exact comparison to a known object. This view needs some enhancements as verification thoroughness is probably proportional to the individuality of the object and the degree of generic identification desired. Human faces need detailed shape comparisons for precise identification, but just to say it was human requires less. Hence, verification should probably use a individual set of identification constraints for each refinement on the identity, much as ACRONYM ([BRO81]) uses a specialization of the constraints in its restriction graph for generic representations.

Unfortunately, ACRONYM's mechanism was too simplistic, because its specialization approach required the constraints of the supertype to be a subset of

those of the more specific type, and it avoided the topic of functionality. For example, chairs have a tremendous variety of shapes, but there is no prototype chair model, even given division into functional groupings. If the only common factors were support for back and seat at given heights, sizes and orientations, then a pile of boxes would also be satisfactory, and this would sometimes be an appropriate identification.

There is some overlap between the functions of hypothesis construction and verification, so should verification be a distinct module? The construct and verify sequence follows the classical "generate and test" paradigm. The goal of the construction process is to: (1) find evidence for all model features and (2) assign a reference frame. To prevent (1) from causing a combinatorial explosion, some constraints were applied when searching for image evidence. On the other hand, verification ensures that the whole object satisfies all constraints, including some previously applied. Hence, there could be some shifting of constraint analysis to verification, particularly if hypothesis construction and verification became more of a parallel process (i.e. akin to a Waltz filtering process). Other justifications for reordering the processes is that partial verification at earlier stages may reduce computational requirements. In any case, the implemented process ordering is largely arbitrary, and the justification of having two processes and thesis chapters is that they are separate topics.

Surface verification of partially obscured or partially back-facing surfaces is weak. For these surfaces, only individual summary characteristics were checked, leaving other tests until the surface was combined with others in an assembly. More detailed symbolic comparisons could be made, as in figure 11–8 below. Here, a square is somewhat obscured. Verification could easily show that it was not a circle, and that it is likely to be a square, by comparing descriptions of the boundary. This technique could also be used for the full and partial boundary comparisons, as proposed in the last section, because rotating and comparing symbolic descriptions is faster and easier than creating the predicted boundary path.

Verification assumed that primitive solids were rigid assemblies and this is

Figure 11–8: Partially Obscured Square Verification

obviously not true for most natural objects. Even if they individually were, within-class variation presents problems for the rigid model. This is apparent with faces, whose shape changes between people and expression. This research allowed some variation by comparing only approximate curvature and orientation in constraint (S_2 and R_3), but this is weak and unlikely to generalize properly. Further, flexible surfaces will also have variable, scale-based segmentation, which will lead to difficulties with constraints based on curvature or correspondence. So, more practical constraints will be needed for richer object domains.

A final criticism is over the number of constraints. Given the essential uniformity in character of real objects, there is probably redundancy in the many constraints proposed. However, given the previous criticisms, many may be inappropriate anyway in a more realistic object domain.

Original Contributions

This chapter introduces original research in two topics:

- Verification was extended to cover fully visible and partially obscured surfaces and assemblies in 3D scenes. This required a computational description of how object location, external occlusion and self-occlusion affect appearance.

- Constraints that help guarantee the existence and identity of rigid structures were formulated and implemented. The constraints were based on the physical properties of surfaces and their appearance.

Chapter 12

Discussion and Conclusions

In previous chapters, the structure and details of an object recognition system based on surface information was motivated, developed and evaluated. This chapter ties these results together by a sequential presentation of the theory's performance on the test images. Special emphasis is given to novel aspects of the research different from those presented in section 1.3. Section 2 discusses these results as a whole and presents the major criticisms and possible direct extensions. Finally, section 3 summarizes the key contributions of the research related the different topics examined in chapters 2-11.

12.1 Several Examples Discussed in Detail

Appendix A shows the two test scenes used in the evaluation of these theories by the IMAGINE program. The key features of the pictures are:

- Test image 1: PUMA Robot Assembly in Trash Can
  - variety of surface shapes and curvature classes forming solids
  - flexibly connected solids
  - partially and completely self-obscured structure
  - externally obscured structure

- structure broken up by occlusion
- intermingled objects

- Test image 2: Chair and Trash Can
  - laminar surfaces
  - concave surfaces
  - symmetric surfaces
  - narrow cylinders
  - multiple invocable objects

Input data

The input data for this analysis was the set of depth and orientation values for the scene, organized as a set of images, and the labeled segmentation boundaries. The segmentation boundaries were found by hand and the depth and orientation values were interpolated within each region from a few values measured by hand. The full data for each scene is shown in Appendix A. For test scene 1, figure A-1 shows the original scene. Figure A-2 shows the depth information coded so that dark means further away. Figures A-3, A-4 and A-5 show the x,y and z component of the unit surface orientation vectors, where brighter means more positive. Figure A-6 shows the segmented surface patches with an identifier assigned to each region. Figure A-7 shows the occlusion boundaries, figure A-8 shows the orientation discontinuity boundaries and figure A-9 (scene 1) shows the curvature discontinuity boundaries. Figures A-10 through A-17 show the same for test scene 2. From these pictures, one can get the general impression of the data and three dimensional character of the scene. It is clearly a lot richer than that of the usual edge-detected picture. It is also denser and probably smoother, too.

Object Representation

The three major object models used in this analysis are the robot, chair and trash can. (These models required definition of 25 SURFACEs and 14 ASSEMBLYs.) The models record:

- the definition of individual surfaces,
- how the surfaces and substructures are linked to form an assembly,
- the invocation network structure detailing the connections over which invocation plausibility flows,
- the direct evidence structure showing acceptable ranges for object properties, and
- additional constraints on the identification of structures.

The complete model definitions are given in appendix B. Also included are synthesised images of each object in a nominal position.

The interesting aspects of the objects are:

- robot: flexible subobject connections, symmetric base, mixed surface shoulder, curved surfaces, complexity
- chair: laminar surfaces, concave surfaces, thin cylindrical legs, symmetric features, concave segmentations, rigidly attached substructures
- trash can: laminar surfaces, symmetry, concave surfaces, simplicity

Surface Hypotheses

The explicit surface hypothesis formation process makes surface hypotheses starting from the segments in the surface image. Additionally, surface reconstruction takes place provided boundary extensions can be linked and surface

compatibility can be ensured. This reconstruction fills in nicks and gaps in individual surfaces and merges pairs of surfaces whose boundaries connect. Because the reconstruction is based on 3D surface image data, it is more reliable than previous work that used only 2D image boundaries. Figures 12–1 and 12–2 show the boundaries of surfaces reconstructed by these processes. The first one shows a multiply segmented planar surface is reconstructed and the second figure shows a separated curved surface reconstructed.

Surface Clusters

The surface hypotheses are then aggregated to form surface clusters, which are object level, identity-independent representations for the solids in the scene. The goal of this process is to create a blob-like solid that encompasses all and only the features associated with a single object. Primitive surface clusters are formed from all surfaces directly connected. Surface clusters are also formed for equivalent depth groups and depth merged surface clusters. This creates larger contexts within which partially self-obscured structure or subcomponents can be found. Here, the examples show only those surface clusters that contain the modeled objects. Figure 12–3 shows the key surface clusters from image 2. The chair also has individual surface clusters for the seat/back group and each of the individual legs, because the seat obscures the tops of the legs and thus isolates them. There are also surface clusters for the other objects, which are not shown. Figure 12–4 shows the surface cluster for the robot lower arm assembly and figure 12–5 shows that for the trash can, as seen in scene 1. There were 18 and 19 surface clusters formed for the test images respectively.

3D Structure Description

Structures then acquire descriptions: length, curvature and relative orientation are computed for curves and surface area, elongation, curvature, curvature axis direction and relative surface orientation are computed for surfaces. (Other properties are also computed.) Because the surface data contains three dimensional

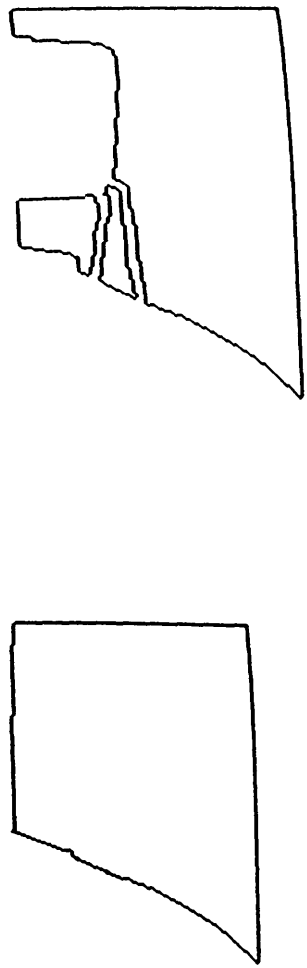


Figure 12–1: Wall Panel From Image 2

Figure 12–2: Trash Can Back From Image 1

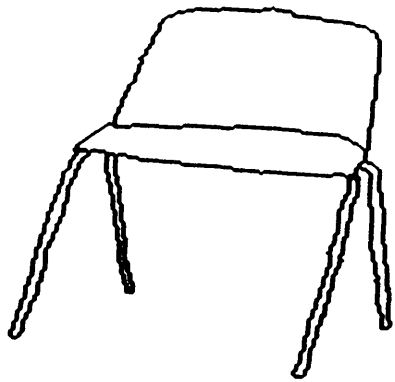


Figure 12–3: Surface Cluster for Chair (Scene 2)

Figure 12–4: Surface Cluster for Robot Lower Arm (Scene 1)

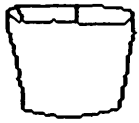


Figure 12–5: Surface Cluster for Trash Can (Scene 1)

information, it is possible to estimate these properties directly instead of having to reconstruct them from image projections. Some of the better examples from the test scenes are given in the tables below. (Boundary segment numbers are from figures 8–3 and 8–4.)

Model Invocation

Model invocation links the identity-independent processing to the model-directed processing by selecting candidate models for further consideration. This process is necessary for both computational efficiency and visual competence, as scene objects will seldom be seen as identical to model objects (because of errors and generic simplifications).

Models are invoked when they acquire sufficiently high plausibility. Plausibility values are calculated in a network of nodes, where there is one node for each potential identity of each image structure. The nodes are connected accord-

Table 12-1: Boundary Curvature (cm^{-1})

IMAGE	REGION	SEGMENTS	ESTIMATED CURVATURE	TRUE CURVATURE
1	26	2	0.120	0.125
1	9	11,12	0.000	0.000
2	4	7	0.040	0.044
2	7	14	0.071	0.069

Table 12-2: Boundary Length (cm)

IMAGE	REGION	SEGMENTS	ESTIMATED LENGTH	TRUE LENGTH	% ERROR
1	8	3,4,5	51.1	50.0	2
1	8	6	27.3	28.2	3
2	7	14	42.2	45.5	7
2	23	18,19	42.8	45.0	5

Table 12-3: Boundary Inter-segment Angles (radians)

IMAGE	REGION	SEGMENTS	ESTIMATED ANGLE	TRUE ANGLE	ERROR
1	9	11,12 -- 13	1.73	1.70	0.03
1	9	17,18,19,20 -- 11,12	1.45	1.44	0.01
2	4	6 -- 7	1.56	1.44	0.12
2	7	12 -- 13	1.46	1.44	0.02

Table 12–4: Absolute Surface Area (cm^2)

TEST IMAGE	IMAGE REGION	PLANAR OR CURVED	ESTIMATED AREA	TRUE AREA	% ERROR
1	9	C	1085	1081	0
1	26	P	165	201	17
2	4	C	1416	1390	2
2	7	C	1074	1081	1

Table 12–5: Surface Curvature (cm^{-1})

TEST IMAGE	IMAGE REGION	ESTIMATED CURVATURE	TRUE CURVATURE
1	8	.127	.111
1	12	0.0	0.0
2	4	-.037	-.044
2	7	.082	.078

Table 12–6: Curved Surface Curvature Axis Orientation

TEST IMAGE	IMAGE REGION	ESTIMATED AXIS	TRUE AXIS	ERROR ANGLE
1	8	(0.0,0.999,0.0)	(0.0,1.0,0.0)	0.02
1	31	(-0.99,-.03,0.11)	(-0.99,0.0,0.1)	0.10
2	4	(0.08,0.99,-0.03)	(0.0,1.0,0.0)	0.09
2	16	(-0.02,0.99,0.07)	(0.0,1.0,0.0)	0.08

Table 12–7: Inter-Surface Angles (radians)

TEST IMAGE	IMAGE REGION	ESTIMATED ANGLE	TRUE ANGLE	ERROR
1	12,18	1.53	1.57	0.04
1	12,31	1.60	1.57	0.03
1	17,22	1.56	1.57	0.01
2	4,9	4.69	4.71	0.02

ing to generic and component relationships between model identities and image contexts.

The plausibility of a node is based on direct and indirect evidence. Indirect evidence comes from the plausibility of other associated structures (usually subcomponents, supercomponents) and generic relationships (i.e. supertypes and subtypes). Direct evidence comes from the degree to which the descriptive properties meet identity-dependent constraints. Inhibition comes from competing identities.

The evidence for these structures accumulates within a context appropriate to the type of structure:

- boundaries associate in a surface context
- individual model surfaces are invoked in a surface hypothesis context
- surfaces associate to form objects in a surface cluster context
- objects associate in a surface cluster context

The plausibility of a particular model being an explanation for an image structure accumulates over pathways made explicit by these relationships. The result is a large network of nodes, representing instances of objects in a particular context that converges to a plausibility value based on the above factors.

Table 12–8: Plausibilities for Trashcan Surfaces in Scene 2

STRUCTURE	IDENTITY	PLAUSIBILITY	
A/16	BOTTOM/tcanbot	-0.54	
A/16	OUTER/tcanoutf	-0.54	
A/16	INNER/tcaninf	-0.35	*
B/7	BOTTOM/tcanbot	-0.72	
B/7	OUTER/tcanoutf	0.30	*
B/7	INNER/tcaninf	0.23	

* – true identity

Because a competent visual system will need to process many images, this network will have to reconfigure for each new image. Section 9.3 proposed a method whereby image boundaries dynamically partition clusters of simple processing units to form the equivalent of the nodes in this scheme. The scheme allowed both static model definitions and dynamic image-based network reconfiguration.

The trashcan node network fragment was shown in figure 9–22. This network will be evaluated for the trashcan data from test image 2. Image region 16 corresponds to example region A and image region 7 corresponds to region B. Then, the plausibilities for the surface nodes in this network are listed in table 12–8. The x/y notation means x in the diagram corresponds to y in the true image/model base network.

These provide the plausibilities entering the diagram at the bottom. The portion of the network immediately above this computes the subcomponent evidence. The three circle units compute the plausibility for the three major viewpoints on the trashcan (see figure 9–4). From left to right, the viewpoints and plausibilities are: from below (-0.12), from significantly above (-0.00) and from somewhat above (0.26). The third case is the one seen in the image and has the highest plausibility. The unit above the circles picks the maximum for the final subcomponent evidence. No generic evidence was used here, so the

upper open_cylinder node has no effect. The maximum competing identity for the trashcan had the plausibility (0.08 for the robot body), so some inhibition is applied. Hence, the final plausibility for the trashcan node is 0.25, and this model is invoked.

When applied to the full test scenes, invocation was generally successful. In test image 1, there 24 surface invocations of 475 possible, of which 10 were correct, 10 were justified because of similarity and 4 were incorrect. There were 17 assembly invocations of 252 possible, of which 10 were correct, 4 were justified because of similarity and 3 were incorrect. Test image 2 showed similar good performance.

Hypothesis Completion

Once a model is invoked, hypothesis completion attempts to fully instantiate the model and simultaneously estimate its spatial location. This occurs in several stages. First, invocation suggests potential model-data correspondences, which arise from high plausibility subcomponent associations. The initial configuration of these structures helps estimate the initial reference frame. Figure 12–6 shows the object boundaries for the robot lower arm initial frame estimate superposed over scene 1.

The estimates came from the correspondences:

IMAGE REGION	MODEL SURFACE
12	lsideb
18	ledgea
31	lendb

The measured and estimated coordinate systems are:

	ROTATION	SLANT	TILT	X	Y	Z
MEASURED	5.06	2.23	1.32	26.6	8.8	538
ESTIMATED	5.22	2.35	1.18	28.1	11.1	546

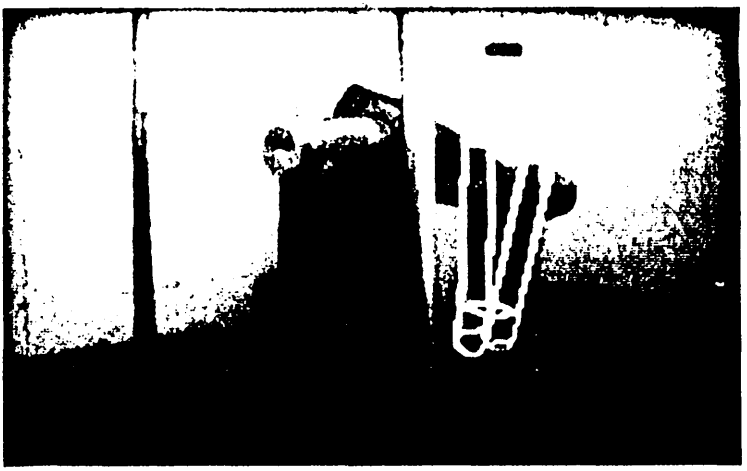


Figure 12–6: Robot Lower Arm in Initial Reference Frame

Even with only a few partially obscured features visible, the extra information in the surface image gives reasonable results.

This estimate allows the process to deduce the visibility status of all substructures, whether fully visible, tangential, back-facing, fully self-obscured or partially self-obscured. The visibility analysis results for the lowerarm assembly are shown below. This analysis is correct (appendix B shows the parts labeling).

STRUCTURE	VISIBILITY STATUS
lsidea	back-facing
lsideb	fully visible
lendb	partially self-obscured (by lendb)
ledgea	fully visible
ledgeb	back-facing
hand	recursively analyzed

In part, this analysis resulted from looking at surface ordering while synthesizing an image of the oriented model. Self-occlusion is determined by comparing

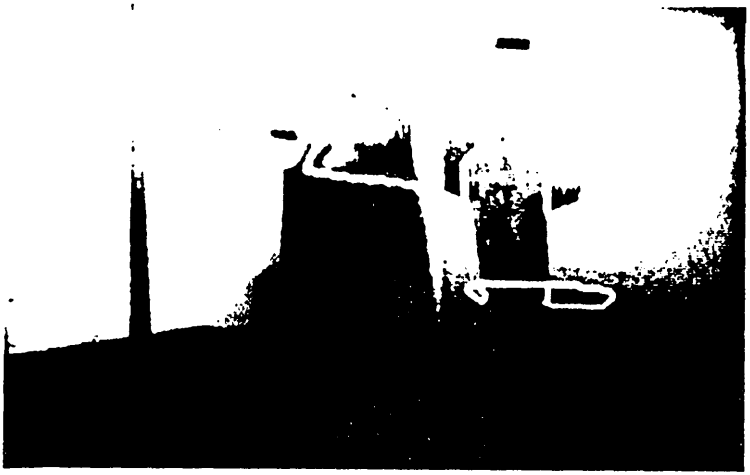


Figure 12–7: Verified Partially Self-Obscured Surfaces for Scene 1

the number of obscured to non-obscured pixels for the front-facing surfaces in the synthetic image. This prediction also allows the program to verify the partially self-obscured surfaces, which was indicated in the data by back-side-obscuring boundaries. Figure 12–7 shows the predicted and verified partially self-obscured surfaces for scene 1.

Image prediction also indicates where one should look in the image to find evidence for features not previously paired to the model. These surfaces or structures must be of the correct type, must have the correct depth and orientation and must be within the correct context.

At this point, the process expects to have found evidence for each visible forward-facing feature. If such cannot be found, then the structures are assumed to be externally obscured. This assumption is verified by showing that all predicted pixels lie behind closer, unrelated obscuring surfaces. Partially obscured (non-self-obscured) surfaces are also verified as being externally obscured. These surfaces are noticed because they have back-side-obscuring boundaries that have

Figure 12–8: Verified Externally Obscured Surfaces in Scene 1

not been explained by self-occlusion analysis. Figure 12–8 shows the verified externally obscured surfaces for the robot in the trash can scene.

This outlines the construction of a completed hypothesis. Two additional points relate to substructures. Verifying missing substructure is a recursive process and is easy given the definition of the objects. Showing the robot hand in scene 1 is obscured decomposes to showing each of the hand's surfaces are obscured.

Second, if the substructures are found, they are linked as a structure to the hypothesis being formed. For rigidly attached structures, this implies comparing the observed reference frame of the object to that predicted by the reference frame of the subobject mapped using the AT map of the model. For scene 1, the reference frame (L) for the robot shoulder small panel assembly (robshldsobj) is:

	ROTATION	SLANT	TILT	X	Y	Z
L:	0.32	2.39	6.10	-16.2	9.8	564

By the model, the transformation for the object's reference frame to the

subobject's (A) is the identity map. The predicted reference frame (LA) for the robot shoulder assembly (robshould) is therefore equal to L (above).

Then, the current best estimate frame for robshould is:

ROTATION	SLANT	TILT	X	Y	Z
0.13	2.30	6.27	-15.7	11.5	562

This is within tolerances, so the small panel is added to the hypothesis.

When a subobject is flexibly joined to the structure, binding of the parameters of flexibility must take place. For the robot upper and lower arm in scene A, the reference frame for the upperarm (U) is:

	ROTATION	SLANT	TILT	X	Y	Z
U:	3.22	2.24	3.14	0.6	17.2	570.

and for the lower arm (L) and its inverse (L') is:

	ROTATION	SLANT	TILT	X	Y	Z
L:	5.22	2.35	1.18	28.1	11.1	546
L':	1.06	2.35	3.26	-132.0	-343.0	404

As the AT map (A) from the model of the robot < *upperasm* > assembly (see appendix B) is the identity map, the predicted FLEX relationship (L'UA) is:

	ROTATION	SLANT	TILT	X	Y	Z
L'UA:	4.35	0.11	0.25	5.1	36.6	2.0

This is then compared to the FLEX relationship given in the model:

ROTATION	SLANT	TILT	X	Y	Z
jnt3	0.0	0.0	0.0	0.0	0.0

The constant values are close enough to the model parameters, so the subobject is bound to the hypothesis, with the flexibility parameters being bound to the other predictions. Hence, jnt3 is bound to 4.35. Because of the lowerarm

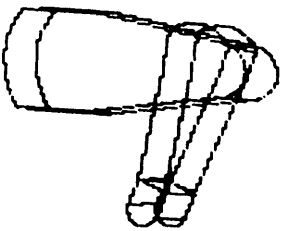


Figure 12–9: Predicted Angle Between Robot Upper and Lower Arms

is partially obscured, its y position estimate is off, which caused the large discrepancy in the L'UA values above. However, it was within tolerances, so the pairing was accepted. Figure 12–9 shows the predicted upper and lower arms at this angle.

Hypothesis Verification

The fully instantiated hypotheses now have their existence and identity verified. For surfaces, existence is guaranteed because they are primitives. Identity requires satisfying the constraints of the model. The constraints implemented were surface shape and arbitrary numerical property constraints (mainly on surface area). Figure 12–10 shows the surfaces in scene 2 that were individually identified and passed verification.

For structures, existence requires all surfaces to be connected. Identity constraints required:

- all visible subfeatures have the correct identity.

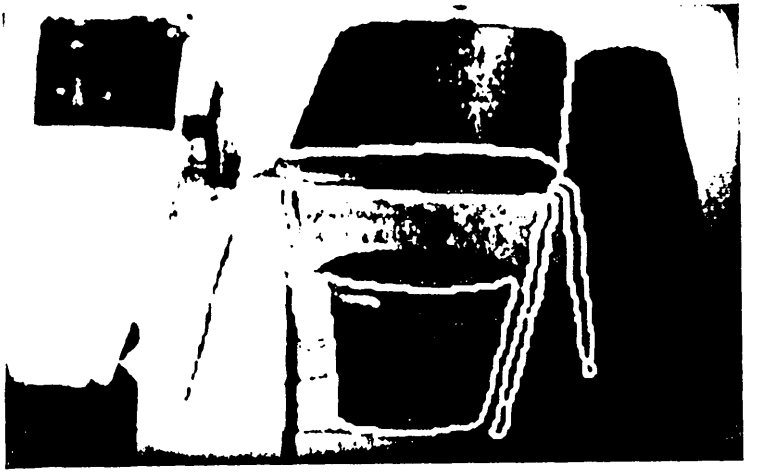


Figure 12–10: Verified Surfaces From Scene 2

- all model features are observed or explained,
- all reference frames are compatible,
- all visible features have the correct orientation and placement,
- predicted surface adjacency holds,
- image evidence is used only once, and
- features supposed to be self-obscured are observed as such.

All true instances of the objects in the test scenes met these constraints. The only other verified hypotheses arose from surfaces similar to the invoked models or symmetric structures. Figures 12–11 through 12–13 show the verified locations of the recognized assemblies superposed over the original images.

To finish the summary, some implementation details follow. The theory was implemented as the **IMAGINE** program in the C programming language under UNIX (c) (Berkeley 4.2) on a VAX 11/750. The size of the program was

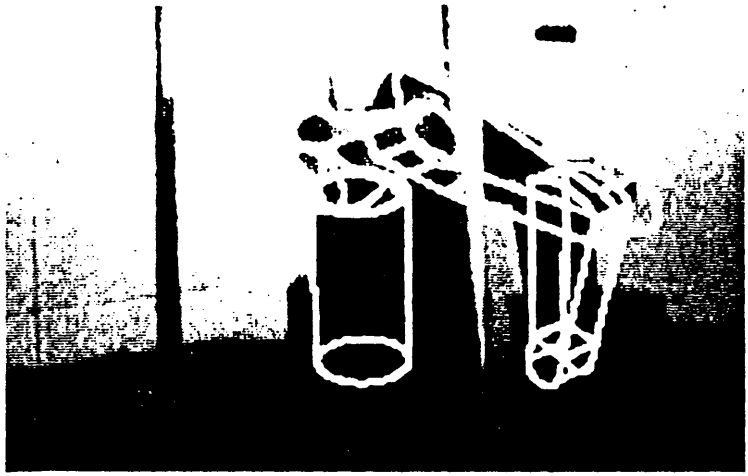


Figure 12–11: Verified Robot in Scene 1

Figure 12–12: Verified Chair and Trash Can in Scene 2

Figure 12–13: Verified Trash Can in Scene 1

about 18,000 lines of somewhat commented code. Execution required about 8 megabytes total, but this included several 256*256 arrays and generous static data structure allocations. Execution times for test image 1 were:

PROCESS	CPU SECONDS
initialization	35
region graph formation	144
surface hypothesizing	290
surface cluster formation	17
description	238
invocation	1022
general hypothesis completion	724
reference frame estimation	397
raycasting	3318
parameter range manipulation	1045
general verification	111
total	7341

Only minor attempts were made to improve performance.

12.2 Summary of Criticisms

This section reviews the key criticisms of the work presented in this thesis, and includes suggestions for extensions and improvements. These points are discussed in greater detail in the relevant chapters.

Input Data

The data used in this research were unrealistic in several respects. Because the depth and orientation values and the segmentation boundaries were hand-derived, they had none of the errors likely to be present in real data. Hence, this approach needs more thorough evaluation. The segmentations also made perfect correspondence with the models, which is unlikely for two reasons: data variation and scale. Data variation, particularly for objects with curved surfaces (e.g. the head) causes shape segmentation boundaries to shift. Further, as the analytic scale changes, segmentation boundaries also move, and segments appear or disappear. The boundary shifting has only minor effect on most processes (except for size description and occlusion analysis), but the addition or deletion of surfaces would be catastrophic. Additionally, because the surface data is only at a single level of scale, more richly structured objects (e.g. trees, parallel slats on a window blind) may not be well detected and described. Clearly scale is a key problem to solve in this area.

Object Modeling

The object representation is too literal. The models should not require exact sizes, nor exact feature placement. The object surfaces should be more notional, designating surface class, curvature, orientation and placement and largely ignore extent.

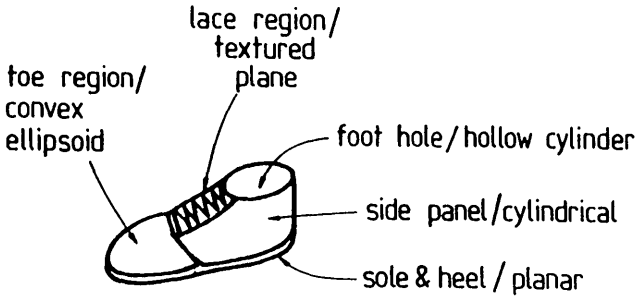


Figure 12–14: Notional Shoe Model

Object representation should also have a more conceptual character that emphasises key distinguishing features and rough geometrical placement, without a full CAD-like model (as used here), though this could be used as a class prototype. An example of such a model for a shoe is in figure 12–14.

As data will occur at unpredictable levels of scale, the models should record the features at a variety of scales. The models should also include other data elements such as references to solids (e.g. generalized cylinders), reflectance, surface shape texture and distinguished axes (e.g. symmetry and elongation). The representation could have used a more constraint-like formulation, as in ACRONYM ([BRO81]), which would allow inequality relationships among features, and also allow easier use of model variables. The thesis largely avoided the problems of generic object representation. Finally, given the amount of information in the model base (appendix B) it is obvious that some automatic method of generating the model base will be needed.

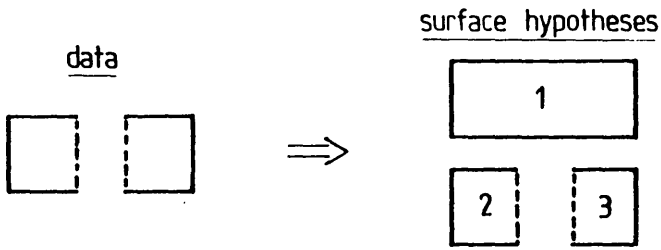


Figure 12–15: Overlapping Surface Hypothesis Formation

Surface Reconstruction

The major problem of the surface reconstruction process is how to resolve the conflict between overlapping surface hypotheses. Figure 12–15 shows a simple case where two data regions give three surface hypotheses. Surfaces 2 and 3 are subsumed by surface 1 if 1 is correct, but if it results from coincidental alignments then 2 and 3 should be kept. Keeping the extra hypotheses causes redundant processing and may lead to duplicated invocations and completed hypotheses.

Surface Cluster Formation

The surface cluster formation process has a similar problem. When one surface cluster overlaps another then a third surface cluster merging the two is created as well. This was to provide a context within which all components of a self-obscured object would appear. The problem is how to control the surface cluster merging process when multiple surface clusters overlap (as is likely in a real scene), which causes a combinatorial growth of surface clusters.

Model Invocation

Invocation currently evaluates a copy of the network in every image context. This is computationally expensive as well as probably wrong when considering the likely number of contexts (100 est.) and the number of models (50000 est.) in a realistic scene. Parallel processing may completely eliminate the computational problem, but there remains the problem of investigating just the relevant contexts. There should probably be a partitioning of the models according to the size of the context, and also some attention focusing mechanism should limit the context within which invocation takes place. This mechanism might apply a rough high-level description of the entire scene and then a coarse-to-fine scale analysis focusing attention to particular regions of interest. The full invocation process may be applied at whatever scale was current to whatever context is being considered. The low level, object-independent vocabulary might be applied everywhere at the current scale of analysis.

Redundant processing will arise because an object will invoke all generalizations of the particular model. The invocations are correct, but the duplication of effort seems wasteful when a direct method could pursue models up and down the generalization hierarchy.

As with the other scale criticisms, invocation only considers a single level of scale for its evidence and the invocation network.

Invocation did not enforce single use of evidence, which could probably have been expressed in the network using other inhibition link types.

Finally, the network, as formulated here, relates subcomponents to objects by identity and grouping, but makes little use of the spatial configuration of features. Some object configuration was implicit in the use of relative orientation information, but more explicit forms could be added. Modeling aspects of the configuration would promote richer networks and help prune coincidental groupings.

Hypothesis Completion

The major criticism of the hypothesis completion process is its literality. In particular, it expects to be able to find evidence for all features, which is probably neither fully necessary, nor likely to be possible in practice. This deficiency arose because the recognition model proposed was incapable of declaring when enough evidence has been accumulated, and so required hypothesis completion to acquire as much evidence as possible. The complete evidence goal for hypothesis completion will become impossible when generic recognition is added. In particular, generic models are likely to be incomplete, and so full model-data correspondences will not be possible.

Literality also appeared in the dependence on the metrical relationships in the body model (in particular, the surface sizes and boundary placements). These were used for predicting self-occlusion and for spatially registering the object. While these tasks are important, and should probably be part of a general vision system, they should have a more conceptual and less analytic formulation. This would provide a stronger symbolic aspect to the computation and should also make the process more capable of handling imperfect or generic objects.

The matching process had few "graceful degradation" features. While it looks for surface evidence in several ways, and the placement parameter range tolerated minor estimation errors, major problems like missing surfaces will cause total failure. The segmentation assumptions prevented this for the examples tested here, but scale-dependent analysis will require a solution to this problem. (Scale will require multiple levels of object description, so that the various features that appear at the data will be matchable.)

Verification

As before, the criticisms of verification largely turn about the issues of literality, perfect data and scale. Verification of identity should probably depend solely on the property comparison and not surface prediction and comparison. In particular, detailed surface shape analysis will fail when comparing a specific

object with a generalized model. Instead, verification should simply ensure that all physical constraints implied by the model are met by whatever data have been paired with the model features, which handles the generalisation case. There should also be some weakening of the geometrical constraints to support generalizations.

Recognition

The most significant limitation of the recognition model proposed in chapter 4 is the absence of scale analysis. Here, objects have different conceptual descriptions according to the relevance of a feature at a given scale, and recognition will then have to match data within a scale-dependent range of models. Another extension would be to match other non-metrical data, such as reflectance and texture, and other shape information, such as axes of alignment.

The criticism of the models and matching being too metrical applies here. A more relational formulation would probably help. One key problem is there does not seem to be a matching method that neatly combines the strengths of relational graph matching with the geometrical reasoning inherent in the use of body models. Including non-metrical data would force the matching to become more symbolic.

Recognition need not require complete evidence or satisfaction of all constraints, provided none are failed, and the few observed features (e.g. a specific color) are adequate for unique identification in a particular context. However, the implementation here plods along trying to find as much evidence as possible. An object should be recognizable using a minimal set of discriminating features and, provided the set of descriptions is powerful enough to discriminate in a large domain, the recognition process will avoid excessive simplification. Recognition (here) has no concept of context, and so cannot make the simplifications.

It might be possible for a program to compute what tests might falsify a given hypothesis.

However, these epistemological questions aside, the recognition process is effective in producing reasonable descriptions of the observed object.

The evaluation on hand-collected and segmented data did not adequately test whether the methods were applicable to real data.

12.3 Research Contributions

This thesis has tried to work at two levels: proposing and justifying a partial paradigm for high-level vision, while also investigating its individual processes. As a result, the research contributions occur at the two corresponding levels.

The key contributions arising from each of the individual modules in the recognition process are:

1. object modeling

    - a surface patch modeling method based on distinct curvature class patches was developed.
    - criteria for choosing how to group surface patches into assemblies were proposed and used.

2. surface data

    - criteria for segmentation of surface image data into characterizable surface patches were proposed and used.

3. surface hypothesizing

    - surface occlusion cases were analyzed to show what cases occur, how to detect them and how to hypothetically reconstruct the missing data. Because the research used 3D surface image data, the reconstruction is more robust than that based on only 2D image data.

4. surface cluster formation

- a new visual representation, the surface cluster, was proposed as an intermediary between the surface image and the object hypotheses.
- rules for aggregating the surface patches into the surface clusters corresponding to distinct objects were proposed and evaluated.

5. description

- a collection of data description modules that took advantage of the three dimensional character of the raw data were developed. Both boundaries and surfaces were described.

6. model invocation

- a network formulation that incorporated both direct evidence from observed properties and indirect evidence from generic or structural relations was developed and evaluated. The formulation was incremental, had operations that were based on general reasoning rather than strictly visual requirements, had provisions for a low-level, object independent vocabulary, and supported dynamic reconfiguration for analyzing new scenes.

7. hypothesis completion

- new methods for estimating the 3D placement of objects from data associated with surface patches and the inter-surface relationships specified by the object model were developed and evaluated.
- methods for predicting and verifying the visibility of surfaces were analyzed and implemented. These could handle back-facing, tangential and partially or fully self-obscured front-facing structure.
- rules for explaining missing structure as instances of occlusion by external, unrelated structure were developed.
- methods for joining flexibly connected structures and simultaneously estimating the spatial parameters of connection were developed, based on a model of weak unification.

- methods for attempting to completely instantiate hypotheses for both solid and laminar structures were developed

8. verification

- new criteria for verifying the physical existence of a hypothesis were proposed and evaluated.
- criteria for verifying the identity of an object based on surface evidence were proposed and evaluated.

At the paradigm level, the key contribution is the exploration of a full artificial intelligence solution to the problem of recognition, using methods that might lead to general purpose vision systems, rather than to limited practical application systems. While efficiency is ultimately important, competence must come first. In particular, only a few researchers have begun to use full $2\frac{1}{2}$D sketch -like surface data, and the work described in this thesis has attempted to explore properly the whole path from surfaces to objects. While the structure of the solution mirrors classical edge-based recognition processes, surface data has required new definitions of the processes and their interconnections.

The use of direct surface data prompted the creation of a new intermediate visual representation, the surface cluster, which is an object-level, but identity independent solid representation suitable for some vision dependent processes (e.g. robot manipulation or collision avoidance). The research also emphasized the strong distinction, but equally strong dependence between the suggestive "seeing" of model invocation and the model-directed hypothesis instantiation and verification. Finally, the effect of occlusion was considered throughout the visual process, and methods were developed that helped overcome data loss at each stage.

The result of this endeavor was a vision system (IMAGINE) that, starting from surface depth and orientation information for the visible surfaces in the scene, could produce an identity-independent segmentation of the objects in the scene, describe their three dimensional properties, select models to explain the

image data, methodically pair the data to model features (while extracting the object's spatial position and explaining missing features arising from occlusion or object position), and verify the existence and identity of the instantiated hypotheses, for non-polyhedral solids, laminar structures and flexibly connected structures, without sacrificing a detailed understanding of the objects or their relationships to the scene.

Bibliography

The following abbreviations are used:

IJCAI - International Joint Conference on Artificial Intelligence

IJCPR - International Joint Conference on Pattern Recognition

NCAI - National Conference on Artificial Intelligence

SRI - Stanford Research Institute

MIT - Massachusetts Institute of Technology

MI - Machine Intelligence

[ABE83] Abe, N., Itho, F., Tsuji, S., *Toward Generation of 3-Dimensional Models of Objects Using 2-Dimensional Figures and Explanations in Language*, IJCAI-8, pp1113-1115, 1983.

[ACK85] Ackley, Hinton and Sejnowski, *A Learning Algorithm for Boltzman Machines*, Cognitive Science, Vol 9, pp147-169, 1985.

[ADL75] Adler, M., *Understanding Peanuts Cartoons*, Dept. of Artificial Intelligence Res. Rpt. #13, Univ. of Edinburgh, 1975.

[AGI73] Agin, G. J., Binford, T. O., *Computer Description of Curved Objects*, Proc. 3rd IJCAI, pp629-640, 1973.

[AGI79] Agin, G. J., *Hierarchical Representation of Three-Dimensional Objects Using Verbal Models*, SRI Tech. Note #182, 1979.

[AIK79] Aikins, J. S., *Prototypes and Production Rules: An Approach to Knowledge Representation For Hypothesis Formation*, Proc. 6th IJCAI, pp 1-3, 1979.

[AMB75] Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M., Popplestone, R. J., *A Versatile System for Computer Controlled Assembly*, Artificial Intelligence, Vol. 6, pp129 - 156, 1975.

[ARB79] Arbib, M. A., *Local Organizing Processes and Motion Schemas in Visual Perception*, in Hayes, et.al., MI-9, pp287-298, 1979.

[ASA84] Asada, H., Brady, M., *The Curvature Primal Sketch*, MIT AI memo #758, 1984.

[BAL81a] Ballard, D. H., *Parameter Networks: Towards a Theory of Low-Level Vision*, IJCAI-7, pp1068-1078, 1981.

[BAL81b] Ballard, D. H., Sabbah, D., *On Shapes*, IJCAI 7, pp607-612, 1981.

[BAL82] Ballard, D. H., Brown, C. M., Computer Vision, Prentice-Hall, 1982.

[BAL85] Ballard, D. H., Tanaka, H., *Transformational Form Perception in 3D: Constraints, Algorithms, Implementations*, Proc. 9th IJCAI, pp964-968, 1985.

[BAR83] Barnard, S.T., Pentland, A.P., *Three-Dimensional Shape from Line Drawing*, IJCAI-8, pp1062-1064, 1983.

[BAR80] Barrow, H., Tenenbaum, J., *Interpreting Line Drawings as Three Dimensional Surfaces*, NCAI-80, Aug 1980.

[BAR74] Barrow, H. G., Burstall, R. M., *Subgraph Isomorphism, Matching Relational Structures and Maximal Cliques*, Edinburgh Dept. of AI DAI-WP-5, Nov 1974.

[BAR71] Barrow, H. G., Popplestone, R. J., *Relational Descriptions in Picture Processing*, Meltzer & Michie, Machine Intelligence 6, pp377-396, 1971.

[BAR72] Barrow, H. G., Ambler, A. P., Burstall, R. M., *Some Techniques for Recognizing Structures in Pictures*, in Frontiers of Pattern Recognition (ed. Watanabe), pp1-29, Academic Press, 1972.

[BAR76] Barrow, H. G., Tenenbaum, J. M., *MSYS: a System for Reasoning About Scenes*, SRI Technical Note 121, 1976.

[BAR78] Barrow, H. G., Tenenbaum, J. M., *Recovering Intrinsic Scene Characteristics from Images*, in Hanson & Riseman (eds), Computer Vision Systems, pp3-26, 1978.

[BER83] Berthod, M., *Global Optimization of a Consistent Labeling*, IJCAI-8, pp1065-1067, 1983.

[BHA83] Bhanu, B., *Recognition of Occluded Objects*, IJCAI-8, pp1136-1138, 1983.

[BIN71] Binford, T. O., *Visual Perception by Computer*, IEEE Conf. on Systems and Control, 1981.

[BIN81] Binford, T. O., *Inferring Surfaces from Images*, AI Vol. 17, pp205-244, 1981.

[BIN82] Binford, T. O., *Survey of Model-Based Image Analysis Systems*, Int. J. of Robotics Research, Vol 1, #1, pp18-64, 1982.

[BLA84] Blake, A., *Reconstructing a Visible Surface*, Proc. 3rd NCAI, pp23-26, 1984.

[BLA85] Blake, A., *Specular Stereo*, Proc. 9th IJCAI, pp 973-976, 1985.

[BOI81] Boissonnat, J. D., Faugeras, O. D., *Triangulation of 3D Objects*, IJCAI-7, pp658-660, 1981.

[BOL83] Bolles, R.C., Horaud, P., Hannah, M. J., *3DPO: a Three-Dimensional Part Orientation System*, IJCAI-8, pp1116-1120, 1983.

[BOL80] Bolles, R., *Locating Partially Visible Objects: The Local Feature Focus Method*, NCAI 80 & SRI TN #223, Aug 1980.

[BOL81] Bolles, R. C., Fischler, M. A., *A RANSAC-Based Approach to Model Fitting and its Application to Finding Cylinders in Range Data*, IJCAI 7, pp637-643, 1981.

[BRA84a] Brady, M., Ponce, J., Yuille, A., Asada, H., *Describing Surfaces*, Proc. 2nd Int. Symp. on Robotics Research, 1984.

[BRA84b] Brady, M., Asada, H., *Smoothed Local Symmetries and Their Implementation*, Int. Journal of Robotics Research, Vol 3, #3, 1984.

[BRA83] Brady, M., Yuille, A., *An Extremum Principle for Shape from Contour*, IJCAI-8, pp969-972, 1983.

[BRI70] Brice, C. R., Fennema, C. L., *Scene Analysis Using Regions*, Artificial Intelligence, Vol. 1, pp205-226, 1970.

[BRO81] Brooks, R.A., *Symbolic Reasoning Among 3D Models and 2D Images*, Stanford AIM-343, STAN-CS-81-861, 1981.

[CAM84] Cameron, S. A. *Modelling Solids in Motion*, PhD Thesis, Dept. of Artificial Intelligence, Univ. of Edinburgh, 1984.

[CER83] Cernuschi-Frias, B., Bolle, R. M., Cooper, D.B., *A New Conceptually Attractive & Computationally Effective Approach to Shape from Shading*, IJCAI-8, pp966-968, 1983.

[CHA79] Chang, N. S., Fu, K. S., *Parallel Parsing of Tree Languages for Syntactic Pattern Recognition*, Pattern Recognition Vol 11 Pg 213, 1979.

[CLO71] Clowes, M. B., *On Seeing Things*, Artificial Intelligence, Vol. 2, pp79-116, 1971.

[CLO80] Clocksin, W., *The Effect of Motion Contrast on Surface Slant and Edge Detection*, AISB-80, July 1980.

[COL81] Coleman, E. N., Jain, R., *Shape from Shading for Surfaces With Texture and Specularity*, IJCAI-7, pp652-657, 1981.

[COW83] Cowie, R. I. D., *The Viewer's Place in Theories of Vision*, IJCAI-8, pp952-958, 1983.

[DAN82] Dane, C., Bajcsy, R., *An Object-Centered Three Dimensional Model Builder*, IJCPR-6, pp348-350, 1982.

[DRE81] Dreschler, L., Nagel, H. H., *Volumetric Model and 3D-Trajectory of a Moving Car Derived from Monocular TV-frame Sequences of a Street Scene*, IJCAI-7, pp692-697, 1981.

[DUD70] DudaR, HartP, *Experiments in Scene Analysis*, SRI report AI group Tech note 20. Project 8259, Jan 1970.

[DUD73] Duda, R., Hart, P. Pattern Classification and Scene Analysis, John Wiley and Sons, New York, 1973.

[FAH80] Fahlman, S. E., *Design Sketch for a Million-Element NETL Machine*, Proc. 1st NCAI, pp249-252, 1980.

[FAH81] , Fahlman, S. E., Touretzky, D. S., van Roggen, W., *Cancellation in a Parallel Semantic Network*, Proc. 7th IJCAI, pp257-263, 1981.

[FAL72] Falk, G., *Interpretation of Imperfect Line Data as a Three-Dimensional Scene*, Artif. Intel. vol 3 (1972), 1972.

[FAU83] Faugeras, O. D., Hebert, M., *A 3-D Recognition and Positioning Algorithm Using Geometric Matching Between Primitive Surfaces*, IJCAI-8, pp996-1002, 1983.

[FAU80] Faugeras, O., *An Optimization Approach for Using Contextual Information in Computer Vision*, NCAI-80, Aug 1980.

[FEL83] Feldman, J. A., Ballard, D. H., *Computing With Connections*, in Human and Machine Vision, Beck, Hope and Rosenfeld (eds), Academic Press, pp 107-155, 1983.

[FEL85] Feldman, J. A., *Four Frames Suffice: a Provisional Model of Vision and Space*, The Behavioral and Brain Sciences, Vol 8, #2, pp265-289, 1985.

[FIS85a] Fisher, R. B., *Discontinuity Segmentation of Curves and Surfaces*, in preparation.

[FIS85b] Fisher, R. B., Orr, M. J. L., *Spatial Configurations in the Model Invocation Process*, submitted to Image and Vision Computing, 1985.

[FIS86] Fisher, R. B., *Recognizing Objects Using Surface Information and Object Models*, submitted to 2nd IEE Int. Conf. on Image Processing, 1986.

[FIS83] Fisher, R. B., *Using Surfaces and Object Models to Recognize Partially Obscured Objects*, IJCAI-8, pp989-995.

[FRE77] Freuder, E. C., *A Computer System for Visual Recognition Using Active Knowledge*, IJCAI 5, pp671-677, 1977.

[GAR82] Garibotto, G., Tosini, R., *Description and Classification of 3D Objects*, IJCPR-6, pp0833-835, 1982.

[GRI81] Grimson, W. E. L., From Images to Surfaces: a Computational Study of the Human Early Visual System, The MIT Press, 1981.

[GUZ67] Guzman, A., *Decomposition of a Visual Scene Into Bodies*, MIT AI memo #139, 1967.

[GUZ68] Guzman, A., *Decomposition of a Visual Scene into Three-Dimensional Bodies*, Proc. Fall Joint Computer Conference, pp291-304, 1968.

[HAN78a] Hanson, A., Riseman, E., *Segmentation of Natural Scenes*, in Hanson & Riseman (eds), Computer Vision Systems, 1978.

[HAN78b] Hanson, A., Riseman, E., *VISIONS: a Computer System for Interpreting Scenes*, in Hanson & Riseman (eds), Computer Vision Systems, pp303-333, 1978.

[HAN80] Hannah, M., *Bootstrap Stereo*, NCAI-80, Aug 1980.

[HEB82] Hebert, M., Ponce, J., *A New Method for Segmenting 3D Scenes Into Primitives*, IJCPR-6, pp836-838, 1982.

[HIN76] Hinton, G., *Using Relaxation to Find a Puppet*, AISB-76, 1976.

[HIN81] Hinton, G., *A Parallel Computation that Assigns Canonical Object-based Frames of Reference*, IJCAI-7, pp683-685, 1981.

[HIN83] Hinton, G. E., Sejnowski, T., J, *Optimal Perceptual Inference*, IEEE Computer and Pattern Recognition Conf., pp448-453, 1983.

[HIN85] Hinton, G. E., Lang, K. J., *Shape Recognition and Illusory Connections*, IJCAI 9, pp252-259, 1985.

[HOG83] Hogg, D., *Model-Based Vision: a Program to See a Walking Person*, Image and Vision Computing, Vol. 1 #1, pp5-20, 1983.

[HOG84] Hogg, D. C., *Interpreting Images of a Known Moving Object*, PhD thesis, Univ. of Sussex, 1984.

[HOP84] Hopfield, J. J., *Neurons With Graded Response Have Collective Computational Properties Like Those of Two-State Neurons*, Proc. American Nat. Acad. Sci., pp3088-3092, 1984.

[HOR81] Horn, B. K. P., Schunck, B. G., *Determining Optical Flow*, Artificial Intelligence, Vol 17, pp185-203, 1981.

[HOR75] Horn, B., *Obtaining Shape from Shading Information*, Winston, The Psychology of Computer Vision, pp115-155, 1975.

[HUF71] Huffman, D. A., *Impossible Objects as Nonsense Sentences*, in Meltzer & Michie (eds), Machine Intelligence 6, 1971.

[IKE81] Ikeuchi, K., *Recognition of 3D Objects Using the Extended Gaussian Image*, IJCAI-7, pp595-600, 1981.

[KAN79] Kanade, T., *A Theory of the Origami World*, IJCAI-6, pp454-456, 1979.

[KAN81a] Kanade, T., Asada, H., *Non-contact Visual Three-Dimensional Ranging Devices*, Proc. SPIE - Int. Soc. for Optical Engineering, pp48-53, April 1981.

[KAN81b] Kanade, T., *Recovery of Three-Dimensional Shape of an Object from a Single View*, Artificial Intelligence, vol 17, 409-460, 1981.

[KEN83] Kender, J.R., *Environment Labelings in Low-Level Image Understanding*, IJCAI-8, pp1104-1107, 1983.

[KOE77] Koenderink, J., J. and van Doorn, A. J., *How an ambulent observer can construct a model of the environment from the geometrical structure of the visual inflow*, in Hauske and Butenandt (eds), KYBERNETIK, pp224-247, 1977.

[KOE82] Koenderink, J., J. and van Doorn, A. J., *The Shape of Smooth Objects and the Way Contours End*, Perception, Vol 11, pp129-137, 1982.

[KOS79] Koshikawa, K., *A Polarimetric Approach to Shape Understanding of Glossy Objects*, IJCAI-6, pp493-495, 1979.

[LOW81] Lowe, D. G., Binford, T. O., *The Interpretation of Three-Dimensional Structure from Image Curves*, IJCAI 7, pp613-618, 1981.

[LOW84] Lowe, D., G., Binford, T. O., *Perceptual Organization as a Basis for Visual Recognition*, Proc. NCAI-84, pp255-260, 1984.

[LUX83] Lux, A., Souvignier, V., *PVV - a Goal-Oriented System for Industrial Vision*, IJCAI-8, pp1121-1124, 1983.

[MAC73] Mackworth, A., *Interpreting Pictures of Polyhedral Scenes*, Artificial Intelligence, Vol 14, pp121-137, 1973.

[MAR82] Marr, D., Vision, pubs: W.H. Freeman and Co., 1982.

[MAR78] Marr, D., Nishihara, H. K., *Representation and Recognition of the Spatial Organization of Three Dimensional Shapes*, Proc. Royal Soc., Vol. 200, 1978.

[MAY80] Mayhew, J., Frisby, J., *Computational and Psychophysical Studies Towards a Theory of Human Stereopsis*, AISB-80, July 1980.

[MAY85] Mayhew, J. E. W., *Issues Concerning The Representation of 3D Shape - a Working Paper 1*, Artif. Intel. Vision Research Unit, Univ. of Sheffield, AIVRU # 1, 1985.

[MIL68] Miller, W. F., Shaw, A. C., *Linguistic Methods in Picture Processing - a Survey*, AFIPS/FJCC Vol 33 Part 1 p279, 1968.

[MIN69] Minsky, M., and Papert, S., Perceptrons, MIT Press, 1969.

[MIN75] Minsky, M. *A Framework for Representing Knowledge*, in Winston (ed), The Psychology of Computer Vision, pp 211-277, 1975.

[MOA76] Moayer, B., Fu, K. S., *A Tree System Approach for Fingerprint Pattern Recognition*, IEEE Trans. Comp. Vol C-25 #3, March 1976.

[MOR81] Moravec, H. P., *Rover Visual Obstacle Avoidance*, 7th IJCAI, pp785-790, 1981.

[NAG79] Nagao, M., Matsuyama, T., Mori, H., *Structural Analysis of Complex Aerial Photographs*, IJCAI 6, pp610-616, 1979.

[NAG83] Nagel, H. H., *Constraints for The Estimation of Displacement Vector Fields from Image Sequences*, IJCAI-8, pp945-951, 1983.

[NEV77] Nevatia, R., Binford, T. O., *Description and Recognition of Curved Objects*, AI Vol. 8, pp77-98, 1977.

[OHT79] Ohta, Y., Kanade, T., Sakai, T., *A Production System for Region Analysis*, IJCAI-6, Aug 1979.

[OHT81] Ohta, Y., Maenobu, K., Sakai, T., *Obtaining Surface Orientation from Texels Under Perspective Projection*, IJCAI 7, pp746-751, 1981.

[OSH81] Oshima, M., Shirai, Y., *Object Recognition Using Three-Dimensional Information*, IJCAI 7, pp601-606, 1981.

[OWE80] Owen, D., *Intermediate Descriptions in "POPEYE"*, AISB-80, July 1980.

[PAU76] Paul, J. L., *Seeing Puppets Quickly*, Proc. AISB, pp221-233, 1976.

[PEN82] Pentland, A. P., *Local Computation of Shape*, Proc ECAI, pp199-204, 1982.

[PEN83] Pentland, A. P., *Fractal-Based Description*, IJCAI-8, pp973-981, 1983.

[PER77] Perkins, W. A., *Model-Based Vision System for Scenes Containing Multiple Parts*, IJCAI 5, pp678-684, 1977.

[PIP82] Pipitone, F. J., *A Ranging Camera and Algorithms for 3D Object Recognition*, PhD thesis, EE dept, Rutgers, 1982.

[POP75] Popplestone, R., J., Brown, C. M., Ambler, A. P., Crawford, G. F., *Forming Models of Plane-And-Cylinder Faceted Bodies from Light Stripes*, Proc. 4th IJCAI, pp 664-668, 1975.

[POT83] Potmesil, M., *Generating Models of Solid Objects by Matching 3D Surface Segments*, IJCAI-8, pp1089-1093, 1983.

[PRA79] Prazdny, K., *Motion and Structure from Optical Flow*, IJCAI-6, pp702-704, 1979.

[REQ77] Requicha, A. A. G., Voelcker, H. B., *Constructive Solid Geometry*, Univ. of Rochester, Production Automation Project memo TM-25, 1977.

[RIE83] Rieger J. H., Lawton, D. T., *Sensor Motion and Relative Depth from Difference Fields of Optic Flows*, IJCAI-8, pp1027-1031, 1983.

[ROB65] Roberts, L. G., *Machine Perception of Three-Dimensional Solids*, Tippett, J. T. (ed.), Optical and Electro-Optical Information Processing, MIT Press, Ch. 9, p159-197, 1965.

[ROS72] Rosenfeld, A., Milgram, D. L., *Web Automata and Web Grammars*, Meltzer et al, Machine Intelligence 7, 1972.

[ROS78] Rosenfeld, A., *Iterative Methods in Image Analysis*, Pattern Recog. Vol 10, pg 181, 1978.

[SCH75] Schank, R. C., Abelson, R. P., *Scripts, Plans, and Knowledge*, in Proc. 4th IJCAI, pp 151-157, 1975.

[SEL60] Selfridge, O. G., Neisser, V., *Pattern Recognition by Machine*, Scientific American 203, pp60-68, 1960.

[SHA80] Shapiro, L., Moriarty, J., Mulgaonkar, P., Haralick, R., *Sticks, Plates, and Blobs: a Three-Dimensional Object Representation for Scene Analysis*, NCAI-80, Aug 1980.

[SHI71] Shirai, Y., Suwa, M., *Recognition of Polyhedrons with a Range-Finder*, Proc. 2nd IJCAI, pp80-87, 1971.

[SHI75] Shirai, Y., *Analyzing Intensity Arrays Using Knowledge About Scenes*, Winston, The Psych. of Comp. Vis., pp93-113, ch 3, 1975.

[SHI78] Shirai, Y., *Recognition of Real-World Objects Using Edge Cue*, Hanson & Riseman (eds), Computer Vision Systems, pp353-362, 1978.

[SHN79] Shneier, M., *A Compact Relational Structure Representation*, IJCAI-6, pp818-826, 1979.

[SLO80] Sloman, A., Owen, D., *Why Visual Systems Process Sketches*, AISB-80, July 1980.

[STE79] Stevens, K. A., *Representing and Analyzing Surface Orientation*, in Winston, Brown (eds), Artif. Intel.: An MIT Perspective, vol 2, p101-125, 1979.

[STE81] Stevens, K. A. *The Visual Interpretation of Surface Contours*, in Brady (ed), Computer Vision, pp 47-73, 1981.

[STE83] Stevens, K. A., *The Line of Curvature Constraint and The Interpretation of 3D Shape from Parallel Surface Contours*, IJCAI-8, pp1057-1061, 1983.

[SUG79] Sugihara, K., *Automatic Construction of Junction Dictionaries and Their Exploitation of the Analysis of Range Data*, IJCAI-6, 1979.

[TAN78] Tanimoto, S., *Regular Hierarchical Image and Processing Structures in Machine Vision*, Hanson & Riseman (eds), Computer Vision Systems, 1978.

[TEN73] Tenenbaum, J., *On Locating Objects by Their Distinguishing Features in Multi-Sensory Images*, SRI Tech Note 84 project 1187, Sept 1973.

[TEN74] Tenenbaum, J., Garvey, T., Weyl, S., Wolf, H., *An Interactive Facility for Scene Analysis Research*, SRI report # 87, project 1187, Jan 1974.

[TEN77] Tenenbaum, J. M., Barrow, H. G., *Experiments in Interpretation Guided Segmentation*, AI Vol 8, pp241-274, 1977.

[TER83] Terzopoulos, D., *The Role of Constraints and Discontinuities in Visible-Surface Reconstructions*, IJCAI-8, pp1073-1077, 1983.

[THO83] Thorpe, C., Shafer, S., *Correspondence in Line Drawings of Multiple Views of Objects*, IJCAI-8, pp959-965, 1983.

[TUR74] Turner, K. J., *Computer Perception of Curved Objects Using a Television Camera*, PhD, University of Edinburgh, 1974.

[WAL75] Walts, D, *Understanding Line Drawings of Scenes with Shadows*, Winston, The Psychology of Computer Vision, pp19-91, 1975.

[WES82] Westphal, H., *Photometric Stereo Considering Diffuse Illumination*, IJCPR-6, pp310-312, 1982.

[WIT80] Witkin, A., *A Statistical Technique for Recovering Surface Orientation from Texture in Natural Imagery*, NCAI-80, Aug 1980.

[WIT83a] Witkin, A.P., *Scale-Space Filtering*, IJCAI-8, pp1019-1022, 1983.

[WIT83b] Witkin, A. P., Tenenbaum, J. M., *What Is Perceptual Organization For?*, IJCAI-8, pp1023-1026, 1983.

[WOO79] Woodham, R., *Analyzing Curved Surfaces Using Reflectance Map Techniques*, in Winston & Brown, Artif. Intel.: An MIT Perspective, vol 2, p161-182, 1979.

[YAC79] Yachida, M., Ikeda, M., Tsuji, S., *A Knowledge Directed Line Finder for Analysis of Complex Scenes*, IJCAI 6, pp984-991, 1979.

[YIN81] Yin, B.L., *A Program Which Recognizes Overlapping Objects*, Edinburgh DAI WP #93, 1981.

[YIN84] Yin, B. L., *Combining Vision Verification with a High Level Robot Programming Language*, PhD thesis, University of Edinburgh, 1984.

[YOR81] York, B. W., Hanson, A. R., Riseman, E. M., *3D Object Representation and Matching with B-Splines and Surface Patches*, IJCAI-7, pp648-651, 1981.

[ZAD79] Zadeh, L. A., *Approximate Reasoning Based on Fuzzy Logic*, Proc. 6th IJCAI, pp1004-1010, 1979.

[ZUC77] Zucker, S., Hummel, R., Rosenfeld, A., *An Application of Relaxation Labeling to Line and Curve Enhancement*, IEEE Trans. Comp. Vol C-26, #4, April 1977.

Appendix A

Test Scenes and Data

The full data for each scene is shown below. Figures A–1 and A–10 shows the original test scenes. Figures A–2 and A–11 shows the depth information coded so dark means further away. Figures A–3 through A–5 and Figures A–12 through A–14 show the x,y and z component of the unit surface orientation vector, where brighter means more positive. Figures A–6 and A–15 show the identifier assigned to each region and the overall segmentation boundaries. Figures A–7 and A–16 show the occlusion label boundaries, figures A–8 and A–17 show the orientation discontinuity label boundaries and figure A–9 shows the curvature discontinuity boundaries.

Figure A–1: Test Scene 1

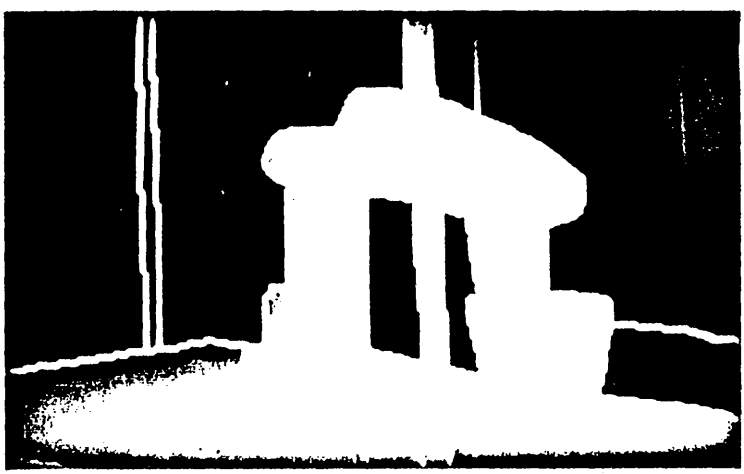


Figure A–2: Test Scene 1 Depth Information

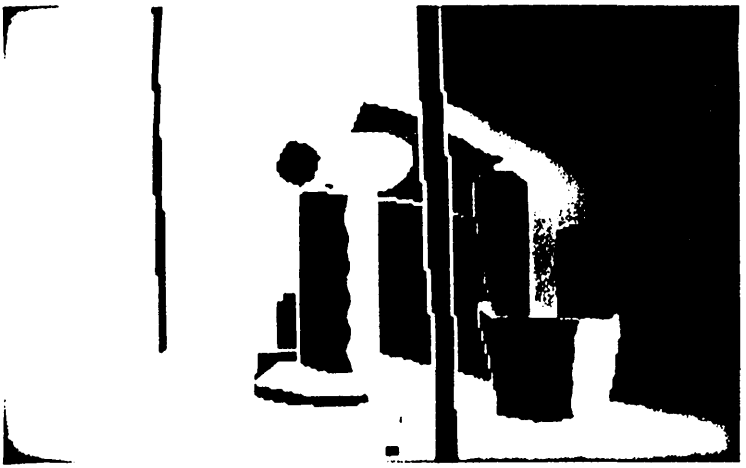


Figure A–3: Test Scene 1 X Component of Surface Orientation

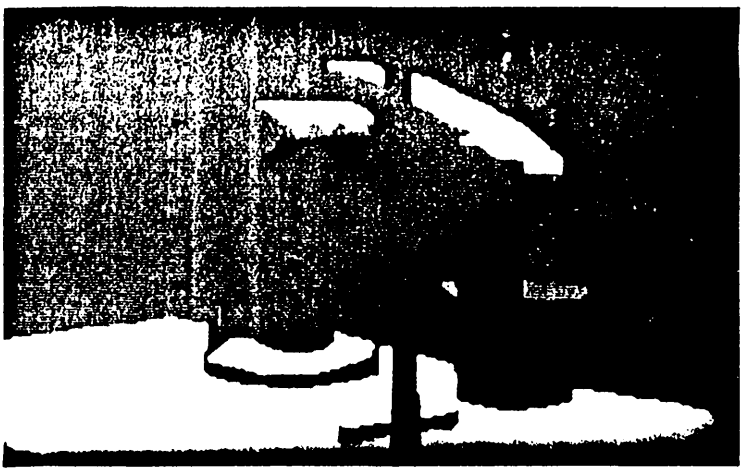


Figure A–4: Test Scene 1 Y Component of Surface Orientation

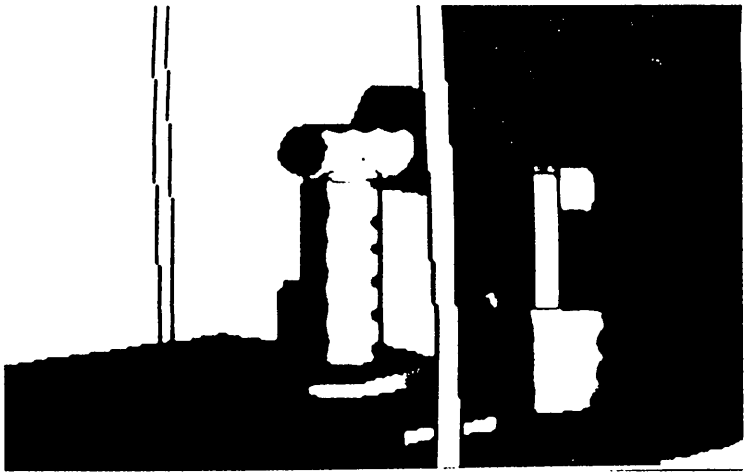


Figure A–5: Test Scene 1 Z Component of Surface Orientation

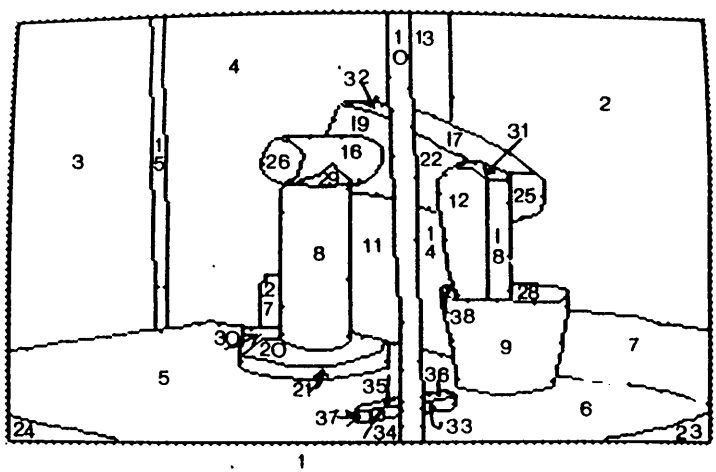


Figure A–6: Test Scene 1 Surface Data Patches with Region Identifiers

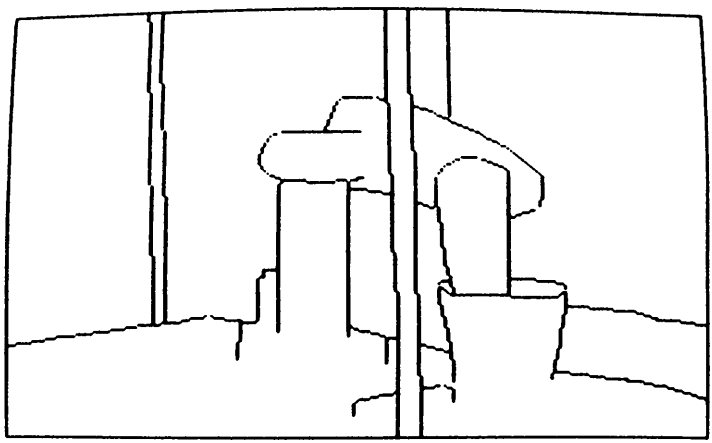


Figure A–7: Test Scene 1 Occlusion Label Boundaries

Figure A–8: Test Scene 1 Orientation Discontinuity Label Boundaries

Figure A–9: Test Scene 1 Curvature Discontinuity Label Boundaries

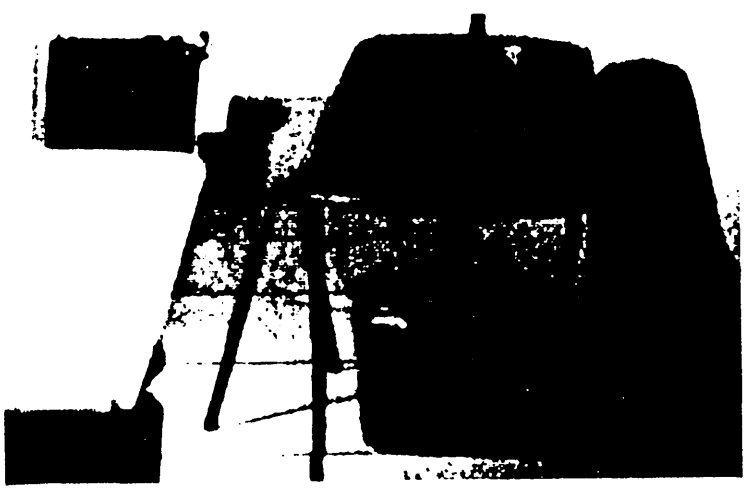


Figure A–10: Test Scene 2

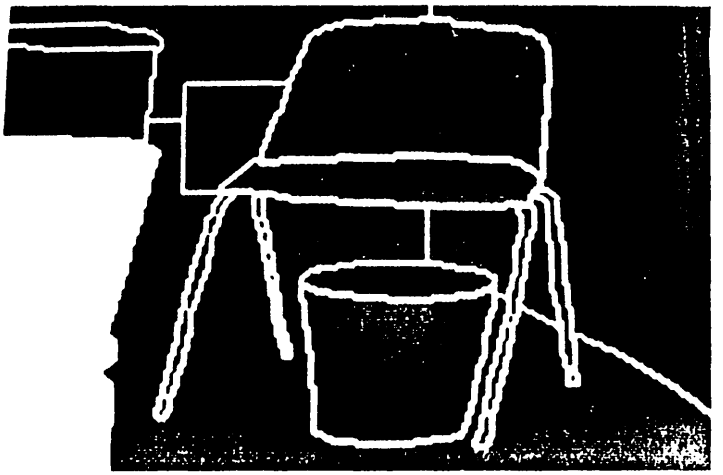


Figure A–11: Test Scene 2 Depth Information

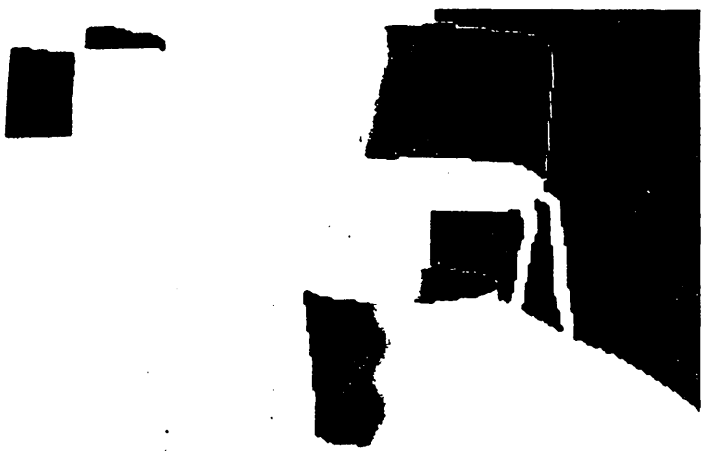


Figure A–12: Test Scene 2 X Component of Surface Orientation

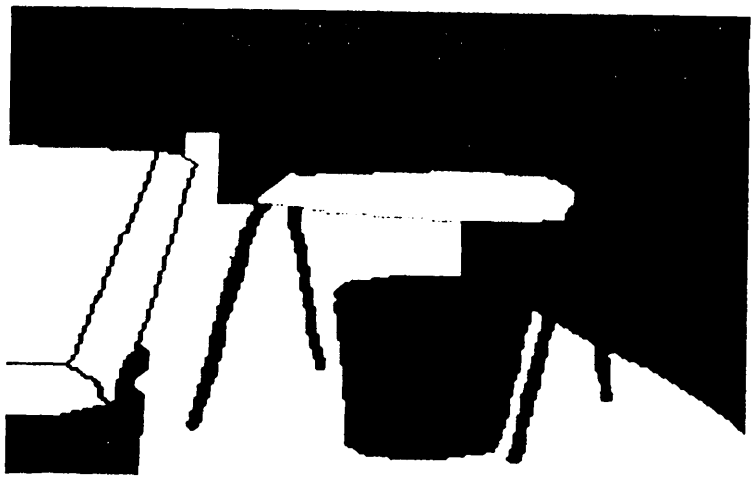


Figure A–13: Test Scene 2 Y Component of Surface Orientation

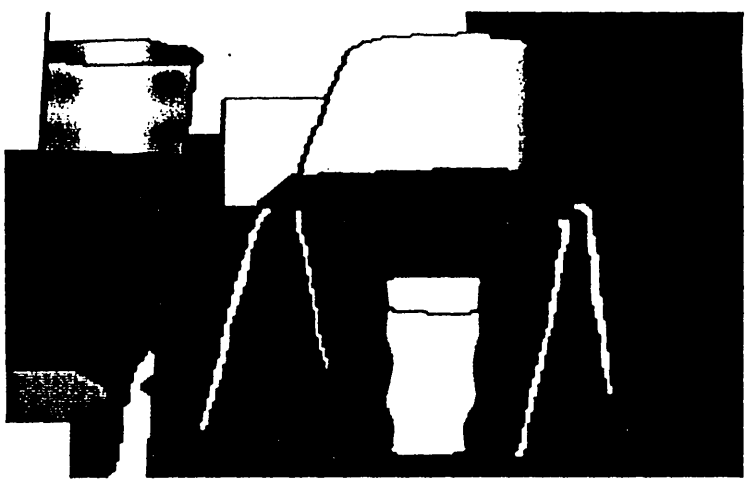


Figure A–14: Test Scene 2 Z Component of Surface Orientation

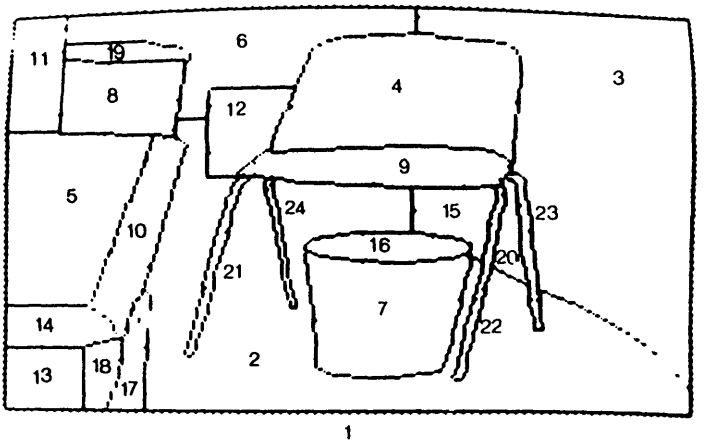


Figure A–15: Test Scene 2 Surface Data Patches with Region Identifiers

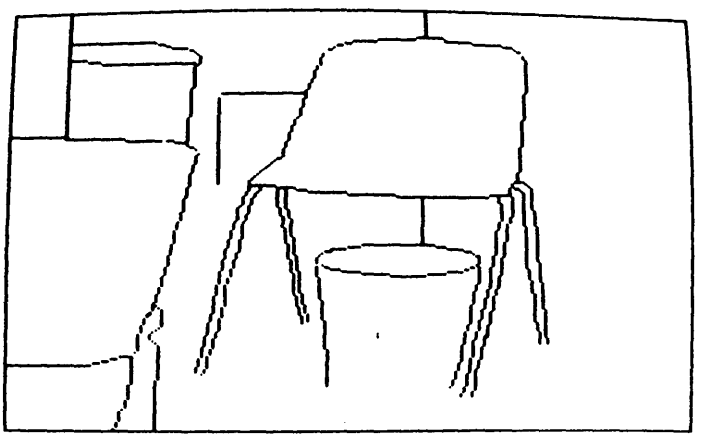


Figure A–16: Test Scene 2 Occlusion Label Boundaries

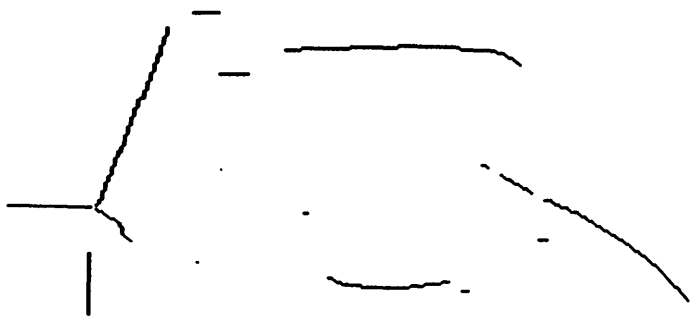


Figure A–17: Test Scene 2 Orientation Discontinuity Label Boundaries

Appendix B

Object Model Definitions

This appendix contains a briefly annotated version of the complete model definitions used in this research. Figures B–1, B–2 and B–3 show cosine shaded images of the models generated from these definitions.

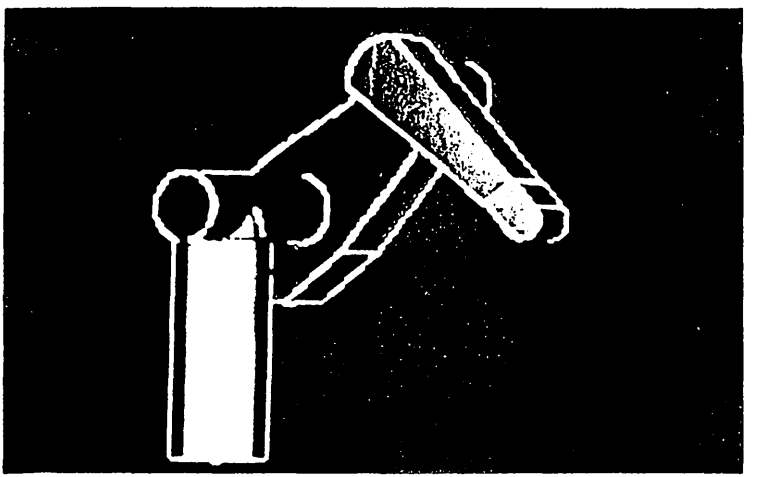


Figure B–1: Robot Model

Chapter 5 describes the contents of the model file in detail. The first section below declares the types of each named entity. OBJTYPE defines ASSEMBLYs, SRFTYPE defines SURFACES and VARTYPE defines variables.

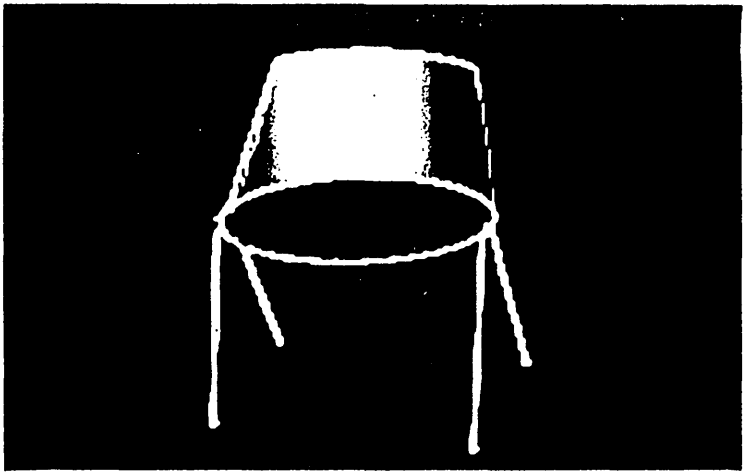


Figure B–2: Chair Model

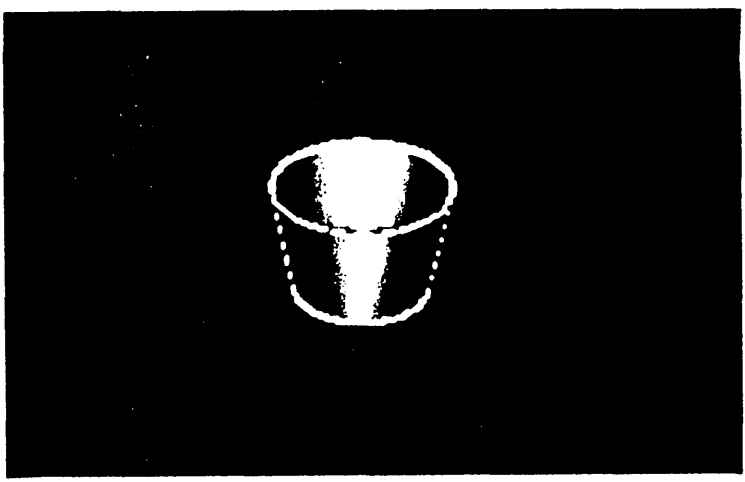


Figure B–3: Trashcan Model

```
TITLE = "combined models";
OBJTYPE robot;
OBJTYPE link;
OBJTYPE upperasm;
OBJTYPE upperarm;
SRFTYPE uside;
SRFTYPE uedges;
SRFTYPE uedgeb;
SRFTYPE uends;
SRFTYPE uendb;
OBJTYPE lowerarm;
SRFTYPE lsidea;
SRFTYPE lsideb;
SRFTYPE ledgea;
SRFTYPE ledgeb;
SRFTYPE lendb;
OBJTYPE hand;
SRFTYPE handend;
SRFTYPE handsidel;
SRFTYPE handsides;
OBJTYPE robbody;
SRFTYPE robbodyside;
OBJTYPE robshould;
OBJTYPE robshldbd;
SRFTYPE robshould1;
SRFTYPE robshould2;
SRFTYPE robshldend;
OBJTYPE robshldsobj;
SRFTYPE robshoulds;
VARTYPE jnt1;
```

```
VARTYPE jnt2;
VARTYPE jnt3;
OBJTYPE chair;
OBJTYPE cseat;
SRFTYPE cseatf;
SRFTYPE cbackf;
SRFTYPE cbackb;
OBJTYPE cleg;
SRFTYPE clegh;
VARTYPE var1;
VARTYPE var2;
VARTYPE var3;
VARTYPE var4;
VARTYPE var5;
OBJTYPE trashcan;
SRFTYPE tcanoutf;
SRFTYPE tcaninf;
SRFTYPE tcanbot;
ENDDEC
```

Then, all the invocation network connections are defined. First the direct associations:

```
SUBCOMPONENT OF robot IS link 1.00;
SUBCOMPONENT OF robot IS robbody 1.00;
SUPERCOMPONENT OF link IS robot 0.10;
SUPERCOMPONENT OF robbody IS robot 0.10;

SUBCOMPONENT OF link IS upperasm 1.00;
```

```
SUBCOMPONENT OF link IS robshould 1.00;
SUPERCOMPONENT OF upperasm IS link 0.10;
SUPERCOMPONENT OF robshould IS link 0.10;

SUBCOMPONENT OF upperasm IS upperarm 0.80;
SUBCOMPONENT OF upperasm IS lowerarm 0.80;
SUPERCOMPONENT OF upperarm IS upperasm 0.10;
SUPERCOMPONENT OF lowerarm IS upperasm 0.10;

SUBCOMPONENT OF upperarm IS uside 0.90;
SUBCOMPONENT OF upperarm IS uendb 0.90;
SUBCOMPONENT OF upperarm IS uends 0.90;
SUBCOMPONENT OF upperarm IS uedges 0.90;
SUBCOMPONENT OF upperarm IS uedgeb 0.90;
SUPERCOMPONENT OF uside IS upperarm 0.10;
SUPERCOMPONENT OF uendb IS upperarm 0.10;
SUPERCOMPONENT OF uends IS upperarm 0.10;
SUPERCOMPONENT OF uedges IS upperarm 0.10;
SUPERCOMPONENT OF uedgeb IS upperarm 0.10;

SUBCOMPONENT OF lowerarm IS lsidea 0.90;
SUBCOMPONENT OF lowerarm IS lsideb 0.90;
SUBCOMPONENT OF lowerarm IS lendb 0.90;
SUBCOMPONENT OF lowerarm IS ledgea 0.90;
SUBCOMPONENT OF lowerarm IS ledgeb 0.90;
SUBCOMPONENT OF lowerarm IS hand 0.90;
SUPERCOMPONENT OF hand IS lowerarm 0.10;
SUPERCOMPONENT OF lsidea IS lowerarm 0.10;
SUPERCOMPONENT OF lsideb IS lowerarm 0.10;
SUPERCOMPONENT OF lendb IS lowerarm 0.10;
SUPERCOMPONENT OF ledgea IS lowerarm 0.10;
```

```
SUPERCOMPONENT OF ledgeb IS lowerarm 0.10;

ASSOCIATION OF upperarm IS lowerarm 1.0;
ASSOCIATION OF lowerarm IS upperarm 1.0;

SUBCOMPONENT OF chair IS cseat 0.90;
SUBCOMPONENT OF chair IS cbackf 0.90;
SUBCOMPONENT OF chair IS cbackb 0.90;
SUBCOMPONENT OF chair IS cleg 0.80;
SUBCOMPONENT OF cseat IS cseatf 0.90;
SUBCOMPONENT OF cleg IS clegh 0.90;
SUPERCOMPONENT OF cseat IS chair 0.10;
SUPERCOMPONENT OF cbackf IS chair 0.10;
SUPERCOMPONENT OF cbackb IS chair 0.10;
SUPERCOMPONENT OF cleg IS chair 0.10;
SUPERCOMPONENT OF cseatf IS cseat 0.10;
SUPERCOMPONENT OF clegh IS cleg 0.10;

SUBCOMPONENT OF robshldbd IS robshould1 0.90;
SUBCOMPONENT OF robshldbd IS robshould2 0.90;
SUBCOMPONENT OF robshldbd IS robshldend 0.90;
SUPERCOMPONENT OF robshould1 IS robshldbd 0.10;
SUPERCOMPONENT OF robshould2 IS robshldbd 0.10;
SUPERCOMPONENT OF robshldend IS robshldbd 0.10;

SUPERCOMPONENT OF robshoulds IS robshldsobj 0.10;
SUBCOMPONENT OF robshldsobj IS robshoulds 0.90;

SUPERCOMPONENT OF robshldbd IS robshould 0.10;
SUPERCOMPONENT OF robshldsobj IS robshould 0.10;
SUBCOMPONENT OF robshould IS robshldsobj 0.90;
```

```
SUBCOMPONENT OF robshould IS robshldbd 0.90;

SUPERCOMPONENT OF robbodyside IS robbody 0.10;
SUBCOMPONENT OF robbody IS robbodyside 0.90;

SUBCOMPONENT OF hand IS handsides 0.90;
SUBCOMPONENT OF hand IS handsidel 0.90;
SUBCOMPONENT OF hand IS handend 0.90;
SUPERCOMPONENT OF handsides IS hand 0.10;
SUPERCOMPONENT OF handsidel IS hand 0.10;
SUPERCOMPONENT OF handend IS hand 0.10;

SUBCOMPONENT OF trashcan IS tcanoutf 0.90;
SUBCOMPONENT OF trashcan IS tcaninf 0.60;
SUBCOMPONENT OF trashcan IS tcanbot 0.40;
SUPERCOMPONENT OF tcanoutf IS trashcan 0.10;
SUPERCOMPONENT OF tcaninf IS trashcan 0.10;
SUPERCOMPONENT OF tcanbot IS trashcan 0.10;
ENDNET
```

Then come the subcomponent groups:

```
SUBCGRP OF robot = robbody link;
SUBCGRP OF link = robshould upperasm;
SUBCGRP OF upperasm = upperarm lowerarm;

SUBCGRP OF upperarm = uside uends uedgeb uedges;
SUBCGRP OF upperarm = uside uendb uedgeb uedges;

SUBCGRP OF lowerarm = lendb lsidea ledgea;
```

```
SUBCGRP OF lowerarm = lendb lsideb ledgea;
SUBCGRP OF lowerarm = lendb lsideb ledgeb;
SUBCGRP OF lowerarm = lendb lsidea ledgeb;

SUBCGRP OF hand = handend handsidea handsidel;

SUBCGRP OF robbody = robbodyside;

SUBCGRP OF robshould = robshldbd robshldsobj;
SUBCGRP OF robshldbd = robshouldi robshldend;
SUBCGRP OF robshldbd = robshould2 robshldend;
SUBCGRP OF robshldsobj = robshoulds;

SUBCGRP OF chair = cseat cbackf cleg;
SUBCGRP OF chair = cseat cbackb cleg;
SUBCGRP OF cseat = cseatf;
SUBCGRP OF cleg = clegh;

SUBCGRP OF trashcan = tcanoutf tcaninf;
SUBCGRP OF trashcan = tcanoutf tcanbot;
SUBCGRP OF trashcan = tcaninf tcanbot tcanoutf;
ENDGRP
```

The final invocation declarations are the constraints that the evidence must meet to contribute to a structure's plausibility.

```
EVIDENCE 1.4 < SURSDA(uside) < 1.7 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(uside) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(uside) < 0.003 WEIGHT 0.5;
```

```
EVIDENCE 0.4 < RELSIZE(uside) < 0.72 WEIGHT 0.5;
EVIDENCE 1000.0 < ABSSIZE(uside) < 2200.0 WEIGHT 0.5;
EVIDENCE 2.0 < SURECC(uside) < 3.2 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(uside) < 0.016 WEIGHT 0.5;
EVIDENCE 0.025 < DCURV(uside) < 0.065 WEIGHT 0.5;
EVIDENCE 0.11 < DCURV(uside) < 0.15 WEIGHT 0.5;
EVIDENCE 10.0 < DCRVL(uside) < 25.0 WEIGHT 0.5;
EVIDENCE 27.0 < DCRVL(uside) < 47.0 WEIGHT 0.5;
EVIDENCE 0.07 < DBRORT(uside) < 0.27 WEIGHT 0.5;
EVIDENCE 1.29 < DBRORT(uside) < 1.67 WEIGHT 0.5;

EVIDENCE 1.45 < SURSDA(uends) < 2.3 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(uends) < 0.003 WEIGHT 0.5;
EVIDENCE 0.09 < DCURV(uends) < 0.17 WEIGHT 0.5;
EVIDENCE 5.0 < DCRVL(uends) < 25.0 WEIGHT 0.5;
EVIDENCE 0 < DBPARO(uends) < 3 WEIGHT 0.2;
EVIDENCE 1.4 < DBRORT(uends) < 1.8 WEIGHT 0.5;
EVIDENCE 1.8 < SURECC(uends) < 2.6 WEIGHT 0.5;
EVIDENCE 130.0 < ABSSIZE(uends) < 250.0 WEIGHT 0.5;
EVIDENCE 0.04 < RELSIZE(uends) < 0.11 WEIGHT 0.5;
EVIDENCE 0.11 < MAXSCURV(uends) < 0.15 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(uends) < 0.015 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(uendb) < 1.65 WEIGHT 0.5;
EVIDENCE 0.08 < RELSIZE(uendb) < 0.16 WEIGHT 0.5;
EVIDENCE 0.035 < MAXSCURV(uendb) < 0.055 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(uendb) < 0.003 WEIGHT 0.5;
EVIDENCE 210.0 < ABSSIZE(uendb) < 430.0 WEIGHT 0.5;
EVIDENCE 2.8 < SURECC(uendb) < 4.0 WEIGHT 0.5;
EVIDENCE 1.47 < DBRORT(uendb) < 1.67 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(uendb) < 3 WEIGHT 0.3;
```

```
EVIDENCE 5.0 < DCRVL(uendb) < 15.0 WEIGHT 0.5;
EVIDENCE 27.0 < DCRVL(uendb) < 37.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(uendb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.025 < DCURV(uendb) < 0.065 WEIGHT 0.5;

EVIDENCE 1.4 < SURSDA(uedges) < 1.8 WEIGHT 0.5;
EVIDENCE 2.8 < SURSDA(uedges) < 3.4 WEIGHT 1.0;
EVIDENCE -0.003 < DCURV(uedges) < 0.003 WEIGHT 0.5;
EVIDENCE 5.0 < DCRVL(uedges) < 25.0 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(uedges) < 3 WEIGHT 0.3;
EVIDENCE 1.47 < DBRORT(uedges) < 1.67 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(uedges) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(uedges) < 0.003 WEIGHT 0.5;
EVIDENCE 0.04 < RELSIZE(uedges) < 0.15 WEIGHT 0.5;
EVIDENCE 140.0 < ABSSIZE(uedges) < 260.0 WEIGHT 0.5;
EVIDENCE 1.8 < SURECC(uedges) < 2.6 WEIGHT 0.5;

EVIDENCE 1.45 < SURSDA(uedgeb) < 1.85 WEIGHT 0.5;
EVIDENCE 2.9 < SURSDA(uedgeb) < 3.1 WEIGHT 1.0;
EVIDENCE -0.003 < MAXSCURV(uedgeb) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(uedgeb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.11 < RELSIZE(uedgeb) < 0.22 WEIGHT 0.5;
EVIDENCE 290.0 < ABSSIZE(uedgeb) < 570.0 WEIGHT 0.5;
EVIDENCE 3.6 < SURECC(uedgeb) < 5.2 WEIGHT 0.5;
EVIDENCE 1.47 < DBRORT(uedgeb) < 1.67 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(uedgeb) < 3 WEIGHT 0.3;
EVIDENCE 5.0 < DCRVL(uedgeb) < 15.0 WEIGHT 0.5;
EVIDENCE 38.0 < DCRVL(uedgeb) < 48.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(uedgeb) < 0.003 WEIGHT 0.5;

EVIDENCE 1.45 < SURSDA(lsidea) < 1.7 WEIGHT 0.5;
```

```
EVIDENCE -0.003 < MAXSCURV(lsidea) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(lsidea) < 0.003 WEIGHT 0.5;
EVIDENCE 0.51 < RELSIZE(lsidea) < 0.65 WEIGHT 0.5;
EVIDENCE 460.0 < ABSSIZE(lsidea) < 910.0 WEIGHT 0.5;
EVIDENCE 2.3 < SURECC(lsidea) < 3.3 WEIGHT 0.5;
EVIDENCE 1.07 < DBRORT(lsidea) < 1.47 WEIGHT 0.5;
EVIDENCE 1.37 < DBRORT(lsidea) < 1.77 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(lsidea) < 3 WEIGHT 0.3;
EVIDENCE 3.6 < DCRVL(lsidea) < 24.0 WEIGHT 0.5;
EVIDENCE 32.8 < DCRVL(lsidea) < 54.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(lsidea) < 0.015 WEIGHT 0.5;
EVIDENCE 0.06 < DCURV(lsidea) < 0.12 WEIGHT 0.5;

EVIDENCE 1.45 < SURSDA(lsideb) < 1.7 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(lsideb) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(lsideb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.51 < RELSIZE(lsideb) < 0.65 WEIGHT 0.5;
EVIDENCE 460.0 < ABSSIZE(lsideb) < 910.0 WEIGHT 0.5;
EVIDENCE 2.3 < SURECC(lsideb) < 3.3 WEIGHT 0.5;
EVIDENCE 1.07 < DBRORT(lsideb) < 1.47 WEIGHT 0.5;
EVIDENCE 1.37 < DBRORT(lsideb) < 1.77 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(lsideb) < 3 WEIGHT 0.3;
EVIDENCE 3.6 < DCRVL(lsideb) < 24.0 WEIGHT 0.5;
EVIDENCE 32.8 < DCRVL(lsideb) < 54.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(lsideb) < 0.015 WEIGHT 0.5;
EVIDENCE 0.06 < DCURV(lsideb) < 0.12 WEIGHT 0.5;

EVIDENCE 1.35 < SURSDA(lendb) < 2.3 WEIGHT 0.5;
EVIDENCE 1.8 < SURECC(lendb) < 2.9 WEIGHT 0.5;
EVIDENCE 70.0 < ABSSIZE(lendb) < 200.0 WEIGHT 0.5;
EVIDENCE 0.07 < RELSIZE(lendb) < 0.18 WEIGHT 0.5;
```

```
EVIDENCE 0.076 < MAXSCURV(lendb) < 0.106 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(lendb) < 0.016 WEIGHT 0.5;
EVIDENCE 0.97 < DBRORT(lendb) < 2.17 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(lendb) < 3 WEIGHT 0.3;
EVIDENCE 4.0 < DCRVL(lendb) < 13.0 WEIGHT 0.5;
EVIDENCE 13.0 < DCRVL(lendb) < 27.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(lendb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.07 < DCURV(lendb) < 0.14 WEIGHT 0.5;

EVIDENCE 1.35 < SURSDA(ledgea) < 2.1 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(ledgea) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(ledgea) < 0.003 WEIGHT 0.5;
EVIDENCE 0.26 < RELSIZE(ledgea) < 0.38 WEIGHT 0.5;
EVIDENCE 230.0 < ABSSIZE(ledgea) < 470.0 WEIGHT 0.5;
EVIDENCE 4.6 < SURECC(ledgea) < 6.6 WEIGHT 0.5;
EVIDENCE 1.4 < DBRORT(ledgea) < 1.8 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(ledgea) < 3 WEIGHT 0.3;
EVIDENCE 3.6 < DCRVL(ledgea) < 13.6 WEIGHT 0.5;
EVIDENCE 33.0 < DCRVL(ledgea) < 55.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(ledgea) < 0.003 WEIGHT 0.5;

EVIDENCE 1.4 < SURSDA(ledgeb) < 2.15 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(ledgeb) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(ledgeb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.26 < RELSIZE(ledgeb) < 0.38 WEIGHT 0.5;
EVIDENCE 230.0 < ABSSIZE(ledgeb) < 470.0 WEIGHT 0.5;
EVIDENCE 4.6 < SURECC(ledgeb) < 6.6 WEIGHT 0.5;
EVIDENCE 1.4 < DBRORT(ledgeb) < 1.8 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(ledgeb) < 3 WEIGHT 0.3;
EVIDENCE 3.6 < DCRVL(ledgeb) < 13.6 WEIGHT 0.5;
EVIDENCE 32.0 < DCRVL(ledgeb) < 54.0 WEIGHT 0.5;
```

```
EVIDENCE -0.003 < DCURV(ledgeb) < 0.003 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(handsides) < 1.65 WEIGHT 0.5;
EVIDENCE 3.04 < SURSDA(handsides) < 3.24 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(handsides) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(handsides) < 0.003 WEIGHT 0.5;
EVIDENCE 0.20 < RELSIZE(handsides) < 0.28 WEIGHT 0.5;
EVIDENCE 56.0 < ABSSIZE(handsides) < 76.0 WEIGHT 0.5;
EVIDENCE 1.0 < SURECC(handsides) < 1.3 WEIGHT 0.5;
EVIDENCE 1.47 < DBRORT(handsides) < 1.67 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(handsides) < 3 WEIGHT 0.3;
EVIDENCE 2.7 < DCRVL(handsides) < 13.6 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(handsides) < 0.003 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(handsidel) < 1.65 WEIGHT 0.5;
EVIDENCE 3.04 < SURSDA(handsidel) < 3.24 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(handsidel) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(handsidel) < 0.003 WEIGHT 0.5;
EVIDENCE 0.30 < RELSIZE(handsidel) < 0.38 WEIGHT 0.5;
EVIDENCE 80.0 < ABSSIZE(handsidel) < 110.0 WEIGHT 0.5;
EVIDENCE 1.2 < SURECC(handsidel) < 1.6 WEIGHT 0.5;
EVIDENCE 1.47 < DBRORT(handsidel) < 1.67 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(handsidel) < 3 WEIGHT 0.3;
EVIDENCE 2.7 < DCRVL(handsidel) < 18.5 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(handsidel) < 0.003 WEIGHT 0.5;
EVIDENCE 0.21 < DCURV(handsidel) < 0.25 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(handend) < 1.65 WEIGHT 0.5;
EVIDENCE 3.04 < SURSDA(handend) < 3.24 WEIGHT 0.5;
EVIDENCE 0.21 < MAXSCURV(handend) < 0.25 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(handend) < 0.003 WEIGHT 0.5;
```

```
EVIDENCE 0.32 < RELSIZE(handend) < 0.52 WEIGHT 0.5;
EVIDENCE 96.0 < ABSSIZE(handend) < 136.0 WEIGHT 0.5;
EVIDENCE 1.0 < SURECC(handend) < 1.2 WEIGHT 0.5;
EVIDENCE 1.47 < DBRORT(handend) < 1.67 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(handend) < 3 WEIGHT 0.3;
EVIDENCE 3.6 < DCRVL(handend) < 18.5 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(handend) < 0.003 WEIGHT 0.5;
EVIDENCE 0.21 < DCURV(handend) < 0.25 WEIGHT 0.5;

EVIDENCE 4.5 < SURSDA(robbodyside) < 4.9 WEIGHT 0.5;
EVIDENCE 2.5 < SURSDA(robbodyside) < 3.7 WEIGHT 0.5;
EVIDENCE 0.09 < MAXSCURV(robbodyside) < 0.14 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(robbodyside) < 0.01 WEIGHT 0.5;
EVIDENCE 0.9 < RELSIZE(robbodyside) < 1.1 WEIGHT 0.5;
EVIDENCE 1200.0 < ABSSIZE(robbodyside) < 1600.0 WEIGHT 0.5;
EVIDENCE 1.57 < SURECC(robbodyside) < 3.5 WEIGHT 0.5;
EVIDENCE 1.17 < DBRORT(robbodyside) < 1.97 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(robbodyside) < 3 WEIGHT 0.3;
EVIDENCE 20.0 < DCRVL(robbodyside) < 36.0 WEIGHT 0.5;
EVIDENCE 40.0 < DCRVL(robbodyside) < 60.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(robbodyside) < 0.015 WEIGHT 0.5;
EVIDENCE 0.05 < DCURV(robbodyside) < 0.16 WEIGHT 0.5;

EVIDENCE 1.4 < SURSDA(robshldend) < 1.8 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(robshldend) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(robshldend) < 0.003 WEIGHT 0.5;
EVIDENCE 0.11 < RELSIZE(robshldend) < 0.40 WEIGHT 0.5;
EVIDENCE 156.0 < ABSSIZE(robshldend) < 248.0 WEIGHT 0.5;
EVIDENCE 0.9 < SURECC(robshldend) < 1.5 WEIGHT 0.5;
EVIDENCE 3.04 < DBRORT(robshldend) < 3.24 WEIGHT 0.5;
EVIDENCE 0 < DBPARO(robshldend) < 2 WEIGHT 0.3;
```

```
EVIDENCE 20.1 < DCRVL(robshldend) < 40.0 WEIGHT 0.5;
EVIDENCE 0.08 < DCURV(robshldend) < 0.15 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(robshould1) < 1.65 WEIGHT 0.5;
EVIDENCE 0.105 < MAXSCURV(robshould1) < 0.145 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(robshould1) < 0.01 WEIGHT 0.5;
EVIDENCE 0.55 < RELSIZE(robshould1) < 0.79 WEIGHT 0.5;
EVIDENCE 428.0 < ABSSIZE(robshould1) < 828.0 WEIGHT 0.5;
EVIDENCE 1.5 < SURECC(robshould1) < 3.5 WEIGHT 0.5;
EVIDENCE 1.4 < DBRORT(robshould1) < 1.8 WEIGHT 0.5;
EVIDENCE 0.8 < DBRORT(robshould1) < 1.1 WEIGHT 0.5;
EVIDENCE 0.9 < DBRORT(robshould1) < 1.5 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(robshould1) < 3 WEIGHT 0.3;
EVIDENCE 5.0 < DCRVL(robshould1) < 16.0 WEIGHT 0.5;
EVIDENCE 11.0 < DCRVL(robshould1) < 21.0 WEIGHT 0.5;
EVIDENCE 18.0 < DCRVL(robshould1) < 37.0 WEIGHT 0.5;
EVIDENCE 0.071 < DCURV(robshould1) < 0.15 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(robshould1) < 0.035 WEIGHT 0.5;

EVIDENCE 1.5 < SURSDA(robshould2) < 1.65 WEIGHT 0.5;
EVIDENCE 0.105 < MAXSCURV(robshould2) < 0.145 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(robshould2) < 0.01 WEIGHT 0.5;
EVIDENCE 0.55 < RELSIZE(robshould2) < 0.79 WEIGHT 0.5;
EVIDENCE 428.0 < ABSSIZE(robshould2) < 828.0 WEIGHT 0.5;
EVIDENCE 1.5 < SURECC(robshould2) < 3.5 WEIGHT 0.5;
EVIDENCE 1.4 < DBRORT(robshould2) < 1.8 WEIGHT 0.5;
EVIDENCE 0.8 < DBRORT(robshould2) < 1.1 WEIGHT 0.5;
EVIDENCE 0.9 < DBRORT(robshould2) < 1.5 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(robshould2) < 3 WEIGHT 0.3;
EVIDENCE 5.0 < DCRVL(robshould2) < 16.0 WEIGHT 0.5;
EVIDENCE 11.0 < DCRVL(robshould2) < 21.0 WEIGHT 0.5;
```

```
EVIDENCE 18.0 < DCRVL(robshould2) < 37.0 WEIGHT 0.5;
EVIDENCE 0.071 < DCURV(robshould2) < 0.15 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(robshould2) < 0.035 WEIGHT 0.5;

EVIDENCE 2.8 < SURSDA(robshoulds) < 3.4 WEIGHT 0.5;
EVIDENCE 0.105 < MAXSCURV(robshoulds) < 0.145 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(robshoulds) < 0.003 WEIGHT 0.5;
EVIDENCE 0.9 < RELSIZE(robshoulds) < 1.1 WEIGHT 0.5;
EVIDENCE 70.0 < ABSSIZE(robshoulds) < 130.0 WEIGHT 0.5;
EVIDENCE 2.0 < SURECC(robshoulds) < 4.0 WEIGHT 0.5;
EVIDENCE 1.8 < DBRORT(robshoulds) < 2.6 WEIGHT 0.5;
EVIDENCE 1.5 < DBRORT(robshoulds) < 2.3 WEIGHT 0.5;
EVIDENCE -1 < DBPARO(robshoulds) < 1 WEIGHT 0.3;
EVIDENCE 7.5 < DCRVL(robshoulds) < 35.0 WEIGHT 0.5;
EVIDENCE 0.05 < DCURV(robshoulds) < 0.131 WEIGHT 0.5;



EVIDENCE 0.8 < SURECC(cseatf) < 1.2 WEIGHT 0.5;
EVIDENCE 1390.0 < ABSSIZE(cseatf) < 1790.0 WEIGHT 0.5;
EVIDENCE 0.75 < RELSIZE(cseatf) < 1.05 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(cseatf) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(cseatf) < 0.003 WEIGHT 0.5;
EVIDENCE 1.15 < SURSDA(cseatf) < 2.00 WEIGHT 0.5;
EVIDENCE 4.64 < SURSDA(cseatf) < 4.78 WEIGHT 0.5;
EVIDENCE 0 < DBPARO(cseatf) < 2 WEIGHT 0.3;
EVIDENCE 50.0 < DCRVL(cseatf) < 110.0 WEIGHT 0.5;
EVIDENCE 0.024 < DCURV(cseatf) < 0.044 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(cseatf) < 0.003 WEIGHT 0.5;

EVIDENCE 2.0 < SURECC(cbackf) < 2.8 WEIGHT 0.5;
```

```
EVIDENCE 1400.0 < ABSSIZE(cbackf) < 2100.0 WEIGHT 0.5;
EVIDENCE 0.75 < RELSIZE(cbackf) < 1.05 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(cbackf) < 0.003 WEIGHT 0.5;
EVIDENCE -0.055 < MAXSCURV(cbackf) < -0.035 WEIGHT 0.5;
EVIDENCE 4.64 < SURSDA(cbackf) < 4.78 WEIGHT 0.5;
EVIDENCE 1.10 < DBRORT(cbackf) < 2.00 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(cbackf) < 3 WEIGHT 0.3;
EVIDENCE 16.4 < DCRVL(cbackf) < 34.1 WEIGHT 0.5;
EVIDENCE 40.0 < DCRVL(cbackf) < 90.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(cbackf) < 0.003 WEIGHT 0.5;
EVIDENCE 0.020 < DCURV(cbackf) < 0.068 WEIGHT 0.5;

EVIDENCE 2.0 < SURECC(cbackb) < 2.8 WEIGHT 0.5;
EVIDENCE 1400.0 < ABSSIZE(cbackb) < 2100.0 WEIGHT 0.5;
EVIDENCE 0.46 < RELSIZE(cbackb) < 0.56 WEIGHT 0.5;
EVIDENCE 0.95 < RELSIZE(cbackb) < 1.05 WEIGHT 0.5;
EVIDENCE 1.50 < SURSDA(cbackb) < 1.65 WEIGHT 0.5;
EVIDENCE 0.035 < MAXSCURV(cbackb) < 0.055 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(cbackb) < 0.003 WEIGHT 0.5;
EVIDENCE 1.10 < DBRORT(cbackb) < 2.0 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(cbackb) < 3 WEIGHT 0.3;
EVIDENCE 16.4 < DCRVL(cbackb) < 34.1 WEIGHT 0.5;
EVIDENCE 40.0 < DCRVL(cbackb) < 90.0 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(cbackb) < 0.003 WEIGHT 0.5;
EVIDENCE 0.02 < DCURV(cbackb) < 0.068 WEIGHT 0.5;

EVIDENCE -0.003 < MAXSCURV(clegh) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(clegh) < 0.003 WEIGHT 0.5;
EVIDENCE 0.005 < RELSIZE(clegh) < 0.025 WEIGHT 0.5;
EVIDENCE 0.95 < RELSIZE(clegh) < 1.05 WEIGHT 0.5;
EVIDENCE 40.0 < ABSSIZE(clegh) < 90.0 WEIGHT 0.5;
```

```
EVIDENCE 15.0 < SURECC(clegh) < 90.0 WEIGHT 0.5;
EVIDENCE 1.2 < DBRORT(clegh) < 2.0 WEIGHT 0.5;
EVIDENCE 0 < DBPARO(clegh) < 2 WEIGHT 0.3;
EVIDENCE 35.0 < DCRVL(clegh) < 55.0 WEIGHT 0.5;
EVIDENCE 0.0 < DCRVL(clegh) < 6.5 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(clegh) < 0.003 WEIGHT 0.5;

EVIDENCE 350.0 < ABSSIZE(tcanbot) < 410.0 WEIGHT 0.5;
EVIDENCE 0.16 < RELSIZE(tcanbot) < 0.30 WEIGHT 0.5;
EVIDENCE 1.6 < SURSDA(tcanbot) < 1.8 WEIGHT 0.5;
EVIDENCE 4.48 < SURSDA(tcanbot) < 4.68 WEIGHT 0.5;
EVIDENCE -0.003 < MAXSCURV(tcanbot) < 0.003 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(tcanbot) < 0.003 WEIGHT 0.5;
EVIDENCE 0.8 < SURECC(tcanbot) < 1.2 WEIGHT 0.5;
EVIDENCE 3.0 < DBRORT(tcanbot) < 3.2 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(tcanbot) < 3 WEIGHT 0.3;
EVIDENCE 0.075 < DCURV(tcanbot) < 0.11 WEIGHT 0.5;
EVIDENCE 30.0 < DCRVL(tcanbot) < 40.0 WEIGHT 0.5;

EVIDENCE 2.9 < SURSDA(tcaninf) < 3.3 WEIGHT 0.5;
EVIDENCE 4.48 < SURSDA(tcaninf) < 4.68 WEIGHT 0.5;
EVIDENCE -0.098 < MAXSCURV(tcaninf) < -0.058 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(tcaninf) < 0.015 WEIGHT 0.5;
EVIDENCE 0.40 < RELSIZE(tcaninf) < 0.99 WEIGHT 0.5;
EVIDENCE 980.0 < ABSSIZE(tcaninf) < 1140.0 WEIGHT 0.5;
EVIDENCE 1.4 < SURECC(tcaninf) < 2.0 WEIGHT 0.5;
EVIDENCE 1.3 < DBRORT(tcaninf) < 1.85 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(tcaninf) < 3 WEIGHT 0.3;
EVIDENCE 0.05 < DCURV(tcaninf) < 0.11 WEIGHT 0.5;
EVIDENCE 19.0 < DCRVL(tcaninf) < 39.0 WEIGHT 0.5;
EVIDENCE 25.0 < DCRVL(tcaninf) < 45.0 WEIGHT 0.5;
```

```
EVIDENCE -0.003 < DCURV(tcaninf) < 0.015 WEIGHT 0.5;

EVIDENCE 0.05 < DCURV(tcanoutf) < 0.11 WEIGHT 0.5;
EVIDENCE -0.003 < DCURV(tcanoutf) < 0.003 WEIGHT 0.5;
EVIDENCE 2.9 < SURSDA(tcanoutf) < 3.3 WEIGHT 0.5;
EVIDENCE 4.48 < SURSDA(tcanoutf) < 4.68 WEIGHT 0.5;
EVIDENCE 0.058 < MAXSCURV(tcanoutf) < 0.098 WEIGHT 0.5;
EVIDENCE -0.003 < MINSCURV(tcanoutf) < 0.015 WEIGHT 0.5;
EVIDENCE 0.40 < RELSIZE(tcanoutf) < 0.99 WEIGHT 0.5;
EVIDENCE 980.0 < ABSSIZE(tcanoutf) < 1140.0 WEIGHT 0.5;
EVIDENCE 1.3 < DBRORT(tcanoutf) < 1.85 WEIGHT 0.5;
EVIDENCE 1.4 < SURECC(tcanoutf) < 2.0 WEIGHT 0.5;
EVIDENCE 1 < DBPARO(tcanoutf) < 3 WEIGHT 0.3;
EVIDENCE 19.0 < DCRVL(tcanoutf) < 39.0 WEIGHT 0.5;
EVIDENCE 25.0 < DCRVL(tcanoutf) < 45.0 WEIGHT 0.5;
ENDINV
```

The structural model definitions follow:

```
ASSEMBLY robot =
      robbody AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      link AT ((0.0,50.0,0.0),(0.0,0.0,0.0))
      FLEX ((0.0,0.0,0.0),(0.0,jnt1,3.14159))
      ;

ASSEMBLY link =
      robshould AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      upperasm AT ((0.0,8.0,-19.0),(0.0,0.0,0.0))
      FLEX ((0.0,0.0,0.0),(jnt2,0.0,0.0))
```

```
      ;

ASSEMBLY upperasm =
      upperarm AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      lowerarm AT ((43.5,0.0,0.0),(0.0,0.0,0.0))
      FLEX ((0.0,0.0,0.0),(jnt3,0.0,0.0))
      ;

ASSEMBLY upperarm =
      uside AT ((-17.0,-14.9,-10.0),(0.0,0.0,0.0))
      uside AT ((-17.0,14.9,0.0),(0.0,3.14,1.5707))
      uendb AT ((-17.0,-14.9,0.0),(0.0,1.5707,3.14159))
      uends AT ((44.8,-7.5,-10.0),(0.0,1.5707,0.0))
      uedges AT ((-17.0,-14.9,0.0),(0.0,1.5707,4.7123))
      uedges AT ((-17.0,14.9,-10.0),(0.0,1.5707,1.5707))
      uedgeb AT ((2.6,-14.9,0.0),(0.173,1.5707,4.7123))
      uedgeb AT ((2.6,14.9,-10.0),(6.11,1.5707,1.5707));

SURFACE uside = PO/(0.0,0.0,0.0) BO/LINE
      PO/(19.6,0.0,0.0) BO/LINE
      PC/(61.8,7.4,0.0) BO/CURVE[7.65,0.0,0.0]
      PC/(61.8,22.4,0.0) BO/LINE
      PO/(19.6,29.8,0.0) BO/LINE
      PO/(0.0,29.8,0.0) BO/CURVE[-22.42,0.0,0.0]
      PLANE
      NORMAL AT (10.0,15.0,0.0) = (0.0,0.0,-1.0);

SURFACE uedges = PO/(0.0,0.0,0.0) BO/LINE
      PO/(19.6,0.0,0.0) BO/LINE
      PO/(19.6,10.0,0.0) BO/LINE
      PO/(0.0,10.0,0.0) BO/LINE
```

```
      PLANE
      NORMAL AT (10.0,5.0,0.0) = (0.0,0.0,-1.0);

SURFACE uedgeb = PO/(0.0,0.0,0.0) BO/LINE
      PO/(42.85,0.0,0.0) BO/LINE
      PO/(42.85,10.0,0.0) BCW/LINE
      PO/(0.0,10.0,0.0) BO/LINE
      PLANE
      NORMAL AT (21.0,5.0,0.0) = (0.0,0.0,-1.0);

SURFACE uends = PO/(0.0,0.0,0.0) BCW/LINE
      PO/(10.0,0.0,0.0) BO/CURVE[0.0,0.0,-7.65]
      PO/(10.0,15.0,0.0) BCW/LINE
      PO/(0.0,15.0,0.0) BO/CURVE[0.0,0.0,-7.65]
      CYLINDER[(0.0,7.5,1.51),(10.0,7.5,1.51),7.65,7.65]
      NORMAL AT (5.0,7.5,-6.14) = (0.0,0.0,-1.0);

SURFACE uendb = PO/(0.0,0.0,0.0) BO/LINE
      PO/(10.0,0.0,0.0) BO/CURVE[0.0,0.0,-22.42]
      PO/(10.0,29.8,0.0) BO/LINE
      PO/(0.0,29.8,0.0) BO/CURVE[0.0,0.0,-22.42]
      CYLINDER [(0.0,14.9,16.75),(10.0,14.9,16.75),22.42,22.42]
      NORMAL AT (5.0,15.0,-5.67) = (0.0,0.0,-1.0);

ASSEMBLY lowerarm =
      lsidea AT ((-9.4,-7.7,0.0),(0.0,0.0,0.0))
      lsideb AT ((-9.4,-7.7,8.6),(0.0,3.14,0.0))
      lendb AT ((-9.4,-7.7,0.0),(1.4536,1.5707,1.5707))
      ledgea AT ((-9.4,-7.7,8.6),(0.0,1.5707,4.7123))
      ledgeb AT ((-7.4,9.3,0.0),(6.083,1.5707,1.5707))
      hand AT ((34.6,-3.8,4.3),(0.0,0.0,0.0))
```

```
      ;

SURFACE lsidea = PO/(0.0,0.0,0.0) BO/LINE
      PO/(44.0,0.0,0.0) BN/LINE
      PO/ (44.0,8.6,0.0) BN/LINE
      PO/(2.0,17.0,0.0) BN/CURVE[-10.96,1.29,0.0]
      PLANE
      NORMAL AT (22.0,6.0,0.0) = (0.0,0.0,-1.0);

SURFACE lsideb = PO/(0.0,0.0,0.0) BO/LINE
      PO/(-44.0,0.0,0.0) BN/LINE
      PO/(-44.0,8.6,0.0) BO/LINE
      PO/(-2.0,17.0,0.0) BO/CURVE[10.96,1.29,0.0]
      PLANE
      NORMAL AT (-22.0,6.0,0.0) = (0.0,0.0,-1.0);

SURFACE ledgea = PO/(0.0,0.0,0.0) BO/LINE
      PO/(44.0,0.0,0.0) BN/LINE
      PO/(44.0,8.6,0.0) BO/LINE
      PO/(0.0,8.6,0.0) BO/LINE
      PLANE
      NORMAL AT (22.0,4.3,0.0) = (0.0,0.0,-1.0);

SURFACE ledgeb = PO/(0.0,0.0,0.0) BO/LINE
      PO/(42.8,0.0,0.0) BN/LINE
      PO/ (42.8,8.6,0.0) BO/LINE
      PO/ (0.0,8.6,0.0) BO/LINE
      PLANE
      NORMAL AT (22.0,4.3,0.0) = (0.0,0.0,-1.0);

SURFACE lendb = PO/(0.0,0.0,0.0) BO/CURVE[0.0,0.0,-11.04]
```

```
      PO/(17.0,0.0,0.0) BO/LINE
      PO/(17.0,8.6,0.0) BO/CURVE[0.0,0.0,-11.04]
      PO/(0.0,8.6,0.0) BO/LINE
      CYLINDER [(8.5,0.0,7.04),(8.5,8.6,7.04),11.04,11.04]
      NORMAL AT (8.5,4.3,-4.0) = (0.0,0.0,-1.0);

ASSEMBLY hand =
      handsidel AT ((0.0,-4.3,-4.3),(0.0,0.0,0.0))
      handsidel AT ((0.0,4.3,4.3),(0.0,3.14,1.5707))
      handsides AT ((0.0,-4.3,4.3),(0.0,1.5707,4.71))
      handsides AT ((0.0,4.3,-4.3),(0.0,1.5707,1.5707))
      handend AT ((7.7,-4.3,-4.3),(0.0,1.57,0.0))
      ;

SURFACE handsides = PO/(0.0,0.0,0.0) BO/LINE
      PO/(7.7,0.0,0.0) BCW/LINE
      PO/(7.7,8.6,0.0) BO/LINE
      PO/(0.0,8.6,0.0) BO/LINE
      PLANE
      NORMAL AT (3.8,4.3,0.0) = (0.0,0.0,-1.0);

SURFACE handsidel = PO/(0.0,0.0,0.0) BO/LINE
      PO/(0.0,8.6,0.0) BO/LINE
      PO/(7.7,8.6,0.0) BO/CURVE[3.04,3.04,0.0]
      PN/(12.0,4.3,0.0) BO/CURVE[3.04,-3.04,0.0]
      PO/(7.7,0.0,0.0) BO/LINE
      PLANE
      NORMAL AT (6.0,4.3,0.0) = (0.0,0.0,-1.0);

SURFACE handend = PO/(0.0,0.0,0.0) BO/CURVE[0.0,-3.04,-3.04]
      PN/(0.0,4.3,-4.3) BO/CURVE[0.0,3.04,-3.04]
```

```
      PO/(0.0,8.6,0.0) BCW/LINE
      PO/(8.6,8.6,0.0) BO/CURVE[0.0,3.04,-3.04]
      PN/(8.6,4.3,-4.3) BO/CURVE[0.0,-3.04,-3.04]
      PO/(8.6,0.0,0.0) BCW/LINE
      CYLINDER [(0.0,4.3,0.0),(8.6,4.3,0.0),4.3,4.3]
      NORMAL AT (4.3,4.3,-4.3) = (0.0,0.0,-1.0);

ASSEMBLY robbody =
      robbodyside AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      robbodyside AT ((0.0,0.0,0.0),(0.0,3.14,0.0))
      ;

SURFACE robbodyside = PO/(-9.0,0.0,0.0) BO/CURVE[-6.364,0.0,-6.364]
      PN/(0.0,0.0,-9.0) BO/CURVE[6.364,0.0,-6.364]
      PO/(9.0,0.0,0.0) BN/LINE
      PO/(9.0,50.0,0.0) BO/CURVE[6.364,0.0,-6.364]
      PN/(0.0,50.0,-9.0) BO/CURVE[-6.364,0.0,-6.364]
      PO/(-9.0,50.0,0.0) BN/LINE
      CYLINDER [(0.0,0.0,0.0),(0.0,50.0,0.0),9.0,9.0]
      NORMAL AT (0.0,25.0,-9.0) = (0.0,0.0,-1.0);

ASSEMBLY robshould =
      robshldbd AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      robshldsobj AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      ;

ASSEMBLY robshldbd =
      robshould1 AT ((0.0,8.0,-19.0),(0.0,1.5707,0.0))
      robshould2 AT ((0.0,8.0,-19.0),(0.0,1.5707,3.14159))
      robshldend AT ((0.0,8.0,10.0),(0.0,3.14,0.0))
      ;
```

```
ASSEMBLY robshldsobj =
      robshoulds AT ((0.0,0.0,0.0),(0.0,1.5707,0.0))
      robshoulds AT ((0.0,0.0,0.0),(0.0,1.5707,3.14159))
      ;

SURFACE robshldend = PN/(-8.0,0.0,0.0) BO/CURVE[-5.66,5.66,0.0]
      PN/(0.0,8.0,0.0) BO/CURVE[5.66,5.66,0.0]
      PN/(8.0,0.0,0.0) BO/CURVE[5.66,-5.66,0.0]
      PN/(0.0,-8.0,0.0) BO/CURVE[-5.66,-5.66,0.0]
      PLANE
      NORMAL AT (0.0,0.0,0.0) = (0.0,0.0,-1.0)
      ;

SURFACE robshoulds = PO/(-8.0,0.0,0.0) BO/CURVE[-5.66,0.0,-5.66]
      PN/(0.0,0.0,-8.0) BO/CURVE[5.66,0.0,-5.66]
      PO/(8.0,0.0,0.0) BO/CURVE[0.0,-6.32,-6.32]
      PO/(0.0,8.0,-8.0) BO/CURVE[0.0,-6.32,-6.32]
      CYLINDER [(0.0,0.0,0.0),(0.0,1.0,0.0),8.0,8.0]
      NORMAL AT (0.0,4.0,-8.0) = (0.0,0.0,-1.0);

SURFACE robshould1 = PO/(0.0,-8.0,0.0) BO/CURVE[0.0,-5.66,-5.66]
      PN/(0.0,0.0,-8.0) BO/CURVE[0.0,5.66,-5.66]
      PO/(0.0,8.0,0.0) BN/LINE
      PO/(29.0,8.0,0.0) BO/CURVE[0.0,5.66,-5.66]
      PN/(29.0,0.0,-8.0) BO/CURVE[0.0,-5.66,-5.66]
      PO/(29.0,-8.0,0.0) BN/LINE
      PO/(27.0,-8.0,0.0) BO/CURVE[0.0,-6.32,-6.32]
      PO/(19.0,0.0,-8.0) BO/CURVE[0.0,-6.32,-6.32]
      PO/(11.0,-8.0,0.0) BN/LINE
      CYLINDER [(0.0,0.0,0.0),(1.0,0.0,0.0),8.0,8.0]
```

```
      NORMAL AT (10.0,0.0,-8.0) = (0.0,0.0,-1.0);

SURFACE robshould2 = PO/(0.0,-8.0,0.0) BO/CURVE[0.0,-5.66,-5.66]
      PN/(0.0,0.0,-8.0) BO/CURVE[0.0,5.66,-5.66]
      PO/(0.0,8.0,0.0) BN/LINE
      PO/(-29.0,8.0,0.0) BO/CURVE[0.0,5.66,-5.66]
      PN/(-29.0,0.0,-8.0) BO/CURVE[0.0,-5.66,-5.66]
      PO/(-29.0,-8.0,0.0) BN/LINE
      PO/(-27.0,-8.0,0.0) BO/CURVE[0.0,-6.32,-6.32]
      PO/(-19.0,0.0,-8.0) BO/CURVE[0.0,-6.32,-6.32]
      PO/(-11.0,-8.0,0.0) BN/LINE
      CYLINDER [(0.0,0.0,0.0),(-1.0,0.0,0.0),8.0,8.0]
      NORMAL AT (-10.0,0.0,-8.0) = (0.0,0.0,-1.0);



ASSEMBLY chair =
      cseat AT ((0.0,0.0,0.0),(0.0,1.5707,1.5707))
      SYM ((0.0,0.0,0.0),(var1,0.0,0.0))
      cbackf AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      cbackb AT ((0.0,0.0,0.0),(0.0,3.14,0.0))
      cleg AT ((-22.5,0.0,-5.0),(0.0,0.2619,1.5707))
      SYM ((0.0,0.0,0.0),(0.0,var2,3.14159))
      cleg AT ((22.5,0.0,-5.0),(0.0,0.2619,1.5707))
      SYM ((0.0,0.0,0.0),(0.0,var3,3.14159))
      cleg AT ((-22.5,0.0,0.0),(0.0,0.4364,4.7122))
      SYM ((0.0,0.0,0.0),(0.0,var4,3.14159))
      cleg AT ((22.5,0.0,0.0),(0.0,0.4364,4.7122))
      SYM ((0.0,0.0,0.0),(0.0,var5,3.14159))
      ;
```

```
ASSEMBLY cleg =
      clegh AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      ;

ASSEMBLY cseat =
      cseatf AT ((0.0,0.0,-0.05),(0.0,0.0,0.0))
      cseatf AT ((0.0,0.0,0.05),(0.0,3.14,0.0));

SURFACE cseatf = PN/(-22.5,0.0,0.0) BO/CURVE[-15.91,15.91,0.0]
      PN/(0.0,22.5,0.0) BO/CURVE[15.91,15.91,0.0]
      PN/(22.5,0.0,0.0) BO/CURVE[15.91,-15.91,0.0]
      PN/(0.0,-22.5,0.0) BO/CURVE[-15.91,-15.91,0.0]
      PLANE
      NORMAL AT (0.0,0.0,0.0) = (0.0,0.0,-1.0);

SURFACE cbackf = PO/(-22.5,0.0,0.0) BO/LINE
      PO/(-17.5,29.0,14.14) BO/CURVE[0.0,0.0,22.5]
      PO/(17.5,29.0,14.14) BO/LINE
      PO/(22.5,0.0,0.0) BO/CURVE[15.91,0.0,15.91]
      PN/(0.0,0.0,22.5) BO/CURVE[-15.91,0.0,15.91]
      CYLINDER[(0.0,0.0,0.0),(0.0,29.0,0.0),-22.5,-22.5]
      NORMAL AT (0.0,14.5,22.5) = (0.0,0.0,-1.0);

SURFACE cbackb = PO/(-22.5,0.0,0.0) BO/LINE
      PO/(-17.5,29.0,-14.14) BO/CURVE[0.0,0.0,-22.5]
      PO/(17.5,29.0,-14.14) BO/LINE
      PO/(22.5,0.0,0.0) BO/CURVE[15.91,0.0,-15.91]
      PN/(0.0,0.0,-22.5) BO/CURVE[-15.91,0.0,-15.91]
      CYLINDER[(0.0,0.0,0.0),(0.0,29.0,0.0),22.5,22.5]
      NORMAL AT (0.0,14.5,-22.5) = (0.0,0.0,-1.0);
```

```
SURFACE clegh = PO/(0.5,0.0,0.0) BN/LINE
      PO/(0.5,-45.0,0.0) BO/CURVE[0.3535,0.0,-0.3535]
      PN/(0.0,-45.0,-0.5) BO/CURVE[-0.3535,0.0,-0.3535]
      PO/(-0.5,-45.0,0.0) BN/LINE
      PO/(-0.5,0.0,0.0) BO/CURVE[-0.3535,0.0,-0.3535]
      PN/(0.0,0.0,-0.5) BO/CURVE[0.3535,0.0,-0.3535]
      CYLINDER[(0.0,0.0,0.0),(0.0,-45.0,0.0),0.5,0.5]
      NORMAL AT (0.0,-22.5,-0.5) = (0.0,0.0,-1.0)
      ;

ASSEMBLY trashcan =
      tcanoutf AT ((0.0,0.0,-0.05),(0.0,0.0,0.0))
      tcanoutf AT ((0.0,0.0,0.05),(0.0,3.14,0.0))
      tcaninf AT ((0.0,0.0,0.0),(0.0,0.0,0.0))
      tcaninf AT ((0.0,0.0,0.0),(0.0,3.14,0.0))
      tcanbot AT ((0.0,0.0,0.0),(0.0,1.5707,1.5707))
      tcanbot AT ((0.0,-0.05,0.0),(0.0,1.5707,4.71))
      ;

SURFACE tcanbot = PN/(-11.0,0.0,0.0) BO/CURVE[-7.778,7.778,0.0]
      PN/(0.0,11.0,0.0) BO/CURVE[7.778,7.778,0.0]
      PN/(11.0,0.0,0.0) BO/CURVE[7.778,-7.778,0.0]
      PN/(0.0,-11.0,0.0) BO/CURVE[-7.778,-7.778,0.0]
      PLANE
      NORMAL AT (0.0,0.0,0.0) = (0.0,0.0,-1.0);

SURFACE tcanoutf = PO/(-11.0,0.0,0.0) BN/LINE
      PO/(-14.5,27.0,0.0) BO/CURVE[-10.253,0.0,-10.253]
      PN/(0.0,27.0,-14.5) BO/CURVE[10.253,0.0,-10.253]
      PO/(14.5,27.0,0.0) BN/LINE
      PO/(11.0,0.0,0.0) BO/CURVE[7.778,0.0,-7.778]
```

```
        PN/(0.0,0.0,-11.0) BO/CURVE[-7.778,0.0,-7.778]
        CYLINDER[(0.0,0.0,0.0),(0.0,27.0,0.0),11.0,14.5]
        NORMAL AT (0.0,13.5,-12.75) = (0.0,-0.1285,-0.9917);

    SURFACE tcaninf = PO/(-11.0,0.0,0.0) BN/LINE
        PO/(-14.5,27.0,0.0) BO/CURVE[-10.253,0.0,10.253]
        PN/(0.0,27.0,14.5) BO/CURVE[10.253,0.0,10.253]
        PO/(14.5,27.0,0.0) BN/LINE
        PO/(11.0,0.0,0.0) BO/CURVE[7.778,0.0,7.778]
        PN/ (0.0,0.0,11.0) BO/CURVE[-7.778,0.0,7.778]
        CYLINDER[(0.0,0.0,0.0),(0.0,27.0,0.0),-11.0,-14.5]
        NORMAL AT (0.0,13.5,12.75) = (0.0,0.1285,-0.9917);

    ENDSTR
```

Finally, any additional verification constraints follow:

```
    CONSTRAINT uside MAXSCURV(uside) < 0.05
    CONSTRAINT uside ABSSIZE(uside) < 1900.0
    CONSTRAINT uside ABSSIZE(uside) > 1050.0
    CONSTRAINT uends ABSSIZE(uends) < 250.0
    CONSTRAINT uendb ABSSIZE(uendb) < 430.0
    CONSTRAINT uedges ABSSIZE(uedges) < 260.0
    CONSTRAINT uedges SURECC(uedges) < 3.0
    CONSTRAINT uedgeb ABSSIZE(uedgeb) < 570.0
    CONSTRAINT lsidea ABSSIZE(lsidea) < 910.0
    CONSTRAINT lsidea ABSSIZE(lsidea) > 300.0
    CONSTRAINT lsideb ABSSIZE(lsideb) < 910.0
    CONSTRAINT lsideb ABSSIZE(lsideb) > 300.0
```

```
CONSTRAINT lendb ABSSIZE(lendb) < 200.0
CONSTRAINT ledgea ABSSIZE(ledgea) < 470.0
CONSTRAINT ledgea ABSSIZE(ledgea) > 200.0
CONSTRAINT ledgeb ABSSIZE(ledgeb) < 470.0
CONSTRAINT ledgeb ABSSIZE(ledgeb) > 200.0
CONSTRAINT handsides ABSSIZE(handsides) < 76.0
CONSTRAINT handsidel ABSSIZE(handsidel) < 110.0
CONSTRAINT handend ABSSIZE(handend) < 136.0
CONSTRAINT robbodyside ABSSIZE(robbodyside) < 1600.0
CONSTRAINT robshldend ABSSIZE(robshldend) < 248.0
CONSTRAINT robshldend SURECC(robshldend) < 1.5
CONSTRAINT robshould1 ABSSIZE(robshould1) < 828.0
CONSTRAINT robshould2 ABSSIZE(robshould2) < 828.0
CONSTRAINT robshoulds ABSSIZE(robshoulds) < 130.0
CONSTRAINT cseatf ABSSIZE(cseatf) < 1790.0
CONSTRAINT cseatf SURECC(cseatf) < 1.5
CONSTRAINT cbackf ABSSIZE(cbackf) < 2100.0
CONSTRAINT cbackb ABSSIZE(cbackb) < 2100.0
CONSTRAINT clegh ABSSIZE(clegh) < 90.0
CONSTRAINT clegh SURECC(clegh) > 2.0
CONSTRAINT tcanbot ABSSIZE(tcanbot) < 410.0
CONSTRAINT tcanoutf ABSSIZE(tcanoutf) < 1140.0
CONSTRAINT tcaninf ABSSIZE(tcaninf) < 1140.0
CONSTRAINT robbodyside SURECC(robbodyside) > 2.0
ENDCON
STOP
```

Just IMAGINE